# **PROPOSAL**

**IMPLEMENTASI ALGORITMA *BI-LINEAR SEARCH* UNTUK PENCARIAN KODE BUKU BERBASIS *WEB***

**(Studi Kasus: Perpustakaan SMA Negeri 5 Kendari)**

**Diajukan untuk memenuhi salah satu syarat memperoleh derajat Sarjana Teknik**

****

**VINA OLYVIA YUNITA ISMAIL**

**E1E117054**

**JURUSAN TEKNIK INFORMATIKA**

**FAKULTAS TEKNIK**

**UNIVERSITAS HALU OLEO**

**KENDARI**

**2021**

# **LEMBAR PENGESAHAN**

**Seminar Proposal**

**IMPLEMENTASI ALGORITMA *BI-LINEAR SEARCH* UNTUK PENCARIAN KODE BUKU BERBASIS *WEB***

**(STUDI KASUS :PERPUSTAKAAN SMA NEGERI 5 KENDARI)**

Adalah benar dibuat oleh saya sendiri dan belum pernah dibuat dan diserahkan sebelumnya baik sebagian ataupun seluruhnya, baik oleh saya ataupun orang lain, baik di Universitas Halu Oleo ataupun instansi pendidikan lainnya.

Kendari, Juli 2021

**Vina Olyvia Yunita Ismail**

**NIM. E1E1 17 054**

**Pembimbing I Pembimbing II**

**Sutardi, S.KOM.,MT** **Adha Mashur Sajiah, ST., M.Eng**

**NIP. 19760222 201012 1 001 NIP. 19910623 201803 1 001**

# **DAFTAR ISI**

[HALAMAN SAMPUL i](#_Toc75883203)

[LEMBAR PENGESAHAN ii](#_Toc75883204)

[DAFTAR ISI iii](#_Toc75883205)

[DAFTAR TABEL iv](#_Toc75883206)

[DAFTAR GAMBAR vi](#_Toc75883207)

[BAB I PENDAHULUAN 1](#_Toc75883208)

[1.1. Latar Belakang 1](#_Toc75883209)

[1.2. Rumusan Masalah 3](#_Toc75883210)

[1.3. Batasan Masalah 3](#_Toc75883211)

[1.4. Tujuan Penelitian 3](#_Toc75883212)

[1.5. Manfaat Penelitian 4](#_Toc75883213)

[1.6. Sistematika Penulisan 4](#_Toc75883214)

[1.7. Tinjauan Pustaka 4](#_Toc75883215)

[BAB II LANDASAN TEORI 6](#_Toc75883216)

[2.1. Perpustakaan 6](#_Toc75883217)

[2.2. Buku 6](#_Toc75883218)

[2.3. Algoritma Pencarian (*Search Algorithm*) 6](#_Toc75883219)

[2.4. Algoritma *Bi-linear Search* 7](#_Toc75883220)

[2.4.1. *Pseudo Code* 8](#_Toc75883221)

[2.4.2. Proses Perhitungan Metode *Bi-linear Search* 9](#_Toc75883222)

[2.5. *Hypertext Preprocessor* (PHP) 11](#_Toc75883223)

[2.6. *CodeIgniter* 12](#_Toc75883224)

[2.7. *Database Management System* 12](#_Toc75883225)

[2.8. *MySQL*](#_Toc75883226) 13

[2.9. XAMPP 14](#_Toc75883227)

[2.10. *Flowchart* 15](#_Toc75883228)

[2.11. Metode Perancangan *Unified Modeling Language* (UML) 17](#_Toc75883229)

[BAB III METODOLOGI PENELITIAN 23](#_Toc75883230)

[3.1. Waktu dan Tempat Penelitian 23](#_Toc75883231)

[3.1.1. Waktu 23](#_Toc75883232)

[3.1.2. Tempat Penelitian 23](#_Toc75883233)

[3.2. Metode Pengumpulan Data 23](#_Toc75883234)

[3.3. Metode Pengembangan Sistem 24](#_Toc75883235)

[3.3.1. Permulaan (*Inception*) 24](#_Toc75883236)

[3.3.2. Perluasan/Perencanaan (*Elaboration*) 24](#_Toc75883237)

[3.3.3. Konstruksi (*Construction*) 24](#_Toc75883238)

[3.3.4. Transisi (*Transition*) 24](#_Toc75883239)

[3.4. Analisis Sistem 24](#_Toc75883240)

[3.4.1. Analisis Kebutuhan Fungsional 25](#_Toc75883241)

[3.4.2. Analisis Kebutuhan Non Fungsional 25](#_Toc75883242)

[3.4.2.1. Kebutuhan Perangkat Keras 25](#_Toc75883243)

[3.4.2.2. Kebutuhan Perangkat Lunak 26](#_Toc75883244)

[3.5. Analisis Perancangan Sistem 26](#_Toc75883245)

[3.5.1. Perancangan *Flowchart* 26](#_Toc75883246)

[3.5.2. *Use Case Diagram* 28](#_Toc75883247)

[*3.5.3*  *Class Diagram* 31](#_Toc75883248)

[3.5.4 *Sequence Diagram* 32](#_Toc75883249)

[*3.5.5 Activity Diagram* 31](#_Toc75883248)

[3.5.6 *Perancangan User Interface* 44](#_Toc75883249)

[3.6. Pengujian 48](#_Toc75883245)

[3.6.1 Pengujian *Black Box* 48](#_Toc75883249)

[3.6.2 Pengujian *Efisiensi Waktu* 48](#_Toc75883249)

[DAFTAR PUSTAKA 50](#_Toc75883250)

# **DAFTAR TABEL**

[**Tabel 2. 1 Simbol-Simbol *Flowchart*** 16](#_Toc66076677)

[**Tabel 2. 2 Simbol-Simbol Use *Case*** 17](#_Toc66076678)

[**Tabel 2. 3 Simbol-Simbol *Activity Diagram*** 19](#_Toc66076680)

[**Tabel 2. 4 Simbol-Simbol *Class Diagram*** 20](#_Toc66076681)

[**Tabel 2. 5 Simbol-Simbol *Sequence Diagram*** 21](#_Toc66076682)

[**Tabel 3. 1 *Gannt Chart* Waktu Penelitian**………..……………………………23](#_Toc66076842)

[**Tabel 3. 2 Spesifikasi Perangkat Keras** 26](#_Toc66076843)

[**Tabel 3. 3 Spesifikasi Perangkat Lunak** 26](#_Toc66076844)

[**Tabel 3. 4 Deskripsi *Use* *Case Login*** 28](#_Toc66076845)

[**Tabel 3. 5 Deskripsi *Use Case* Mengelolah Data Anggota** 29](#_Toc66076847)

[**Tabel 3. 6 Deskripsi *Use Case* Mengelolah Data Buku**..………...……………30](#_Toc66076842)

[**Tabel 3. 7 Deskripsi *Use Case* Melihat Katalog Buku** 30](#_Toc66076843)

[**Tabel 3. 8 Deskripsi *Use Case* Mencari Buku** 31](#_Toc66076844)

# **DAFTAR GAMBAR**

[**Gambar 3. 1 *Flowchart System***……………………………………………………...27](#_Toc66077502)

[**Gambar 3. 2 *Use Case Diagram*** 28](#_Toc66077503)

[**Gambar 3. 3 *Class Diagram*** 32](#_Toc66077504)

[**Gambar 3. 4 *Sequence Diagram Login*** 33](#_Toc66077505)

[**Gambar 3. 5 *Sequence Diagram* Menambah Anggota** 34](#_Toc66077506)

[**Gambar 3. 6 *Sequence Diagram* Menambah Buku** 35](#_Toc66077507)

[**Gambar 3. 7 *Sequence Diagram* Melihat Katalog** 36](#_Toc66077508)

[**Gambar 3. 8 *Sequence Diagram* Meminjam Buku** 36](#_Toc66077509)

[**Gambar 3. 9 *Sequence Diagram* Pengembalian Buku** 37](#_Toc66077510)

[**Gambar 3. 10 *Activity Diagram Login*** 38](#_Toc66077511)

[**Gambar 3. 11 *Activity Diagram* Melihat Katalog** 39](#_Toc66077512)

[**Gambar 3. 12 *Activity Diagram* Pinjam Buku** 40](#_Toc66077513)

[**Gambar 3. 13 *Activity Diagram* Tambah Anggota** 41](#_Toc66077514)

[**Gambar 3. 14 *Activity Diagram* Tambah Buku** 42](#_Toc66077515)

[**Gambar 3. 15 *Activity Diagram* Pengembalian Buku** 43](#_Toc66077516)

[**Gambar 3. 16 Tampilan Halaman *Form Login*** 44](#_Toc66077518)

[**Gambar 3. 17 Tampilan Halaman *Menu Home*** 44](#_Toc66077519)

[**Gambar 3. 18 Tampilan Halaman *Menu* Katalog Buku** 45](#_Toc66077520)

[**Gambar 3. 19 Tampilan Halaman *Menu* Pendaftaran Buku** 46](#_Toc66077521)

[**Gambar 3. 20 Tampilan Halaman *Menu* Peminjaman Buku** 47](#_Toc66077522)

[**Gambar 3. 21 Tampilan Halaman *Menu* Pengembalian Buku** 47](#_Toc66077523)

[**Gambar 3. 22 Tampilan Halaman *Menu* Pendaftaran Anggota** 48](#_Toc66077524)

# **BAB I PENDAHULUAN**

* 1. **Latar Belakang**

Teknologi informasi merupakan salah satu teknologi yang sedang berkembang pesat pada saat ini. Dengan kemajuan teknologi informasi, pengaksesan terhadap data atau informasi yang tersedia dapat berlangsung dengan cepat, efisien serta akurat. Peningkatan profesionalisme dan kualitas serta pelayanan yang ditawarkan suatu lembaga merupakan salah satu daya tarik yang menjadi perhatian masyarakat sebagai salah satu kriteria dalam pelayanan masyarakat (Zanuddin, 2016). Penerapan teknologi informasi telah menyebar hampir di semua bidang, tidak terkecuali di bidang perpustakaan. Perpustakaan adalah mencakup suatu ruangan, bagian dari gedung/ bangunan atau gedung tersendiri yang berisi buku buku koleksi, yang diatur dan disusun demikian rupa, sehingga mudah untuk dicari dan dipergunakan apabila sewaktu-waktu diperlukan oleh pembaca (Sataloff dkk., 2017).

Buku adalah kumpulan kertas berisi informasi, tercetak, disusun secara sistematis, dijilid serta bagian luarnya diberi pelindung terbuat dari kertas tebal, karton atau bahan lain (Ii dkk., 2012). Buku dapat ditemukan di area-area perpustakaan di setiap instansi pendidikan. Di perpustakaan buku dipinjamkan dan wajib untuk dikembalikan. Perpustakaan sekarang harus lebih fleksibel mengikuti perkembangan zaman yang ada. Perkembangan dari sisi teknologi salah satunya pencarian kode buku yang memudahkan *staff* untuk mengatur dan mengelola katalog buku di perpustakaan.

Di perpustakaan SMA Negeri 5 Kendari *system* pencarian dan pengelolaan bukunya masih manual. Maka dari itu, dirancanglah sebuah aplikasi berbasis web yaitu sistem informasi pencarian buku dengan menggunakan suatu metode pencarian. Ada begitu banyak jenis algoritma pencarian yang digunakan untuk mencari elemen dari sekumpulan elemen, seperti *sequential search, binary search, interpolation search,* dan *bi-linear search.*

Algoritma *sequential search* dimana pencarian tersebut dilakukan dengan membandingkan setiap elemen larik satu persatu secara beruntun sampai dengan elemen yang dicari ditemukan (Sembiring, 2013). *Interpolation Search* adalah sebuah algoritma atau metode untuk mencari nilai *key* yang diberikan dalam *array* diindeks yang telah diperintahkan oleh nilai – nilai kunci. Metode ini didasari pada proses pencarian nomor telepon pada buku telepon yang mana manusia mencari melalui dengan nilai kunci yang terdapat pada buku. Teknik *searching* ini dilakukan dengan perkiraan letak data (H. Kazwini, 2010). *Binary search* adalah algoritma pencarian untuk data yang terurut. Pencarian dilakukan dengan cara menebak apakah data yang dicari berada ditengah-tengah data, kemudian membandingkan data yang dicari dengan data yang ada ditengah. Bila data yang ditengah sama dengan data yang dicari, berarti data ditemukan. Namun, bila data yang ditengah lebih besar dari data yang dicari, maka dapat dipastikan bahwa data yang dicari kemungkinan berada disebelah kiri dari data tengah dan data disebelah kanan data tengah dapat diabai (Seidl dan Enderle, 2011). Dan algoritma *bi-linear search* yaitu pencarian menyebar di kedua ujung daftar untuk *item* pencarian dalam daftar sehingga skenario kasus terburuknya jauh lebih baik daripada algoritma pencarian lainnya (Sultana dkk., 2017).

Pencarian *linear/sequential* bekerja secara berurutan. Jadi, jika jumlah elemen sangat besar maka waktu eksekusi bertambah dan untuk mencari elemen terakhir, dibuat jumlah perbandingan n. Akibatnya, kompleksitas waktu kasus terburuknya adalah O (n). Hal yang sama terjadi dalam kasus pencarian *Interpolasi*. Dalam hal ini, pencarian Biner menguntungkan. Ini membutuhkan akses data secara acak, memiliki kompleksitas waktu O (log n). Tapi, Pencarian biner membutuhkan lebih banyak ruang dalam tumpukan karena sifatnya yang rekursif. Pencarian *linear* lebih baik daripada pencarian *biner* dalam kasus daftar kecil data. Pencarian *Bi-linear* mengalahkan pencarian *Biner* dengan kompleksitas waktu O (1), ketika item yang dicari ada di posisi terakhir dari *array*. Oleh karena itu, waktu eksekusi pencarian *Bi-linear* sangat cepat dan sangat efisien untuk mencari item dari sejumlah besar *record* (Paira dan Chandra, 2014).

Berdasarkan latar belakang yang telah diuraikan sebelumnya, dalam penelitian ini akan dirancang **“Implementasi Algoritma *Bi-Linear Search* Untuk Pencarian Kode Buku Berbasis *Web.* Studi Kasus : Perpustakaan SMA Negeri 5 Kendari ”.**

* 1. **Rumusan Masalah**

Berdasarkan latar belakang masalah yang diuraikan sebelumnya maka rumusan masalah penelitian ini adalah:

1. Bagaimana merancang sistem informasi pencarian kode buku berbasis *web*?
2. Bagaimana menerapkan algoritma *bi-linear search* pada sistem informasi pencarian kode buku berbasis *web*?
   1. **Batasan Masalah**

Berdasarkan rumusan masalah yang diuraikan diatas maka batasan masalah dalam tugas akhir ini adalah:

1. Penelitian ini dilakukan di Perpustakaan Daerah Kota Kendari Sulawesi Tenggara.
2. Pembuatan sistem informasi pencarian kode buku menggunakan algoritma *bi-linear search.*
3. Sistem informasi pencarian kode buku berbasis *website*.
4. Parameter yang digunakan berupa judul, nama pengarang, dan jumlah buku yang ada.
   1. **Tujuan Penelitian**

Tujuan dari penelitian ini adalah:

* + - 1. Untuk merancang sistem informasi pencarian kode buku berbasis *web.*
      2. Untuk menerapkan algoritma *bi-linear search* dalam sistem informasi pencarian kode buku berbasis *web.*
  1. **Manfaat Penelitian**

Adapun manfaat dari penelitian yang akan dilakukan diharapkan dapat:

1. Agar dapat memudahkan *staff* perpustakakaan dalam mencari kode buku.
2. Agar dapat menjadi acuan tambahan bagi penulis atau peneliti selanjutnya dalam menerapkan algoritma *bi-linear search* terhadap sistem pencarian.
   1. **Sistematika Penulisan**

Sistematika penulisan tugas akhir ini terdiri atas beberapa bagian utama sebagai berikut:

**BAB I PENDAHULUAN**

Merupakan bab pendahuluan yang menguraikan latar belakang masalah, rumusan masalah, batasan masalah, tujuan penelitian, manfaat penelitian, dan sistematika penulisan.

**BAB II LANDASAN TEORI**

Membahas mengenai dasar-dasar teori pendukung yang berhubungan dengan masalah yang diambil dan program aplikasi yang akan digunakan dalam pembangunan system.

**BAB III METODE PENELITIAN**

Membahas mengenai waktu dan tempat penelitian, metode pengumpulan data, metode pembangunan sistem yang digunakan dalam penelitian tugas akhir ini.

## 1.7. Tinjauan Pustaka

Smita Paira, [Sourabh Chandra](https://www.researchgate.net/profile/Sourabh-Chandra?enrichId=rgreq-c6737ed794b317e67ead0b634341da14-XXX&enrichSource=Y292ZXJQYWdlOzI3NjQxMDgzMDtBUzoyMzAxNTk0ODc4NjA3MzZAMTQzMTg4NTcwNjYxOA%3D%3D&el=1_x_5&_esc=publicationCoverPdf), [Sk Safikul Alam](https://www.researchgate.net/profile/Sk-Alam?enrichId=rgreq-c6737ed794b317e67ead0b634341da14-XXX&enrichSource=Y292ZXJQYWdlOzI3NjQxMDgzMDtBUzoyMzAxNTk0ODc4NjA3MzZAMTQzMTg4NTcwNjYxOA%3D%3D&el=1_x_5&_esc=publicationCoverPdf) (2014) dalam penelitianya yang berjudul “Pencarian *Bi Linear* Sesi Baru Pencarian” menjelaskan tentang pengenalan teknik pencarian baru (Pencarian *Bi-linier*), yang dapat beroperasi pada array yang diurutkan dan tidak disortir secara berurutan, namun lebih efisien daripada banyak algoritma pencarian lainnya. Pencarian *Bi-linear* bekerja dari kedua ujung larik. Dalam iterasi pertama, elemen pertama dan terakhir dibandingkan secara bersamaan dengan tombol penelusuran. Jika data ditemukan, metode pencarian berhenti. Jika tidak, elemen kedua dan elemen kedua hingga terakhir dibandingkan secara bersamaan dan seterusnya. Jadi, jumlah langkah / perbandingan yang dibutuhkan adalah n / 2 yaitu setengah dari yang dibutuhkan oleh Pencarian *Linier*. Penelitian ini telah membuktikan Pencarian *Bi-linear* menjadi sangat efisien untuk daftar data yang tidak diurutkan dan diperlukan waktu yang relatif lebih sedikit untuk mencari item dari koleksi rekaman yang besar daripada Pencarian *Linier* biasa. Bahkan dalam daftar yang diurutkan, pencarian *Bi-linear* mengalahkan pencarian *Biner* dengan kompleksitas waktu O (1), ketika item yang dicari ada di posisi terakhir dari *array*. Oleh karena itu, waktu eksekusi pencarian *Bi-linear* sangat cepat dan sangat efisien untuk mencari item dari sejumlah besar *record* (Paira dan Chandra, 2014).

Najma Sultana, Smita Paira, Sourabh Chandra, Sk Safikul Alam (2017) dalam penelitiannya yang berjudul “Sebuah Studi Singkat dan Analisis Tentang Pencarian yang Berbeda” menjelaskan tentang perbandingan pada empat algoritma pencarian tradisional yaitu pencarian *linier*, pencarian *biner*, *interpolasi* dan pencarian lompat. Studi perbandingan pada algoritma ini didasarkan pada fitur, analisis kompleksitas waktu, merit dan demerit masing-masing. Manfaat dari masing-masing dan setiap algoritma sangat dihargai, tetapi tidak dapat mengabaikan kekurangannya. Tetap harus fokus pada kekurangan tersebut untuk memiliki algoritma pencarian yang optimal untuk operasi pencarian yang akurat dan cepat dalam waktu dekat (Sultana dkk., 2017).

# **BAB II LANDASAN TEORI**

1. **Perpustakaan**

Perpustakaan adalah mencakup suatu ruangan, bagian dari gedung/ bangunan atau gedung tersendiri yang berisi buku buku koleksi, yang diatur dan disusun demikian rupa, sehingga mudah untuk dicari dan dipergunakan apabila sewaktu-waktu diperlukan oleh pembaca (Sataloff dkk., 2017)

1. **Buku**

Buku adalah kumpulan kertas atau bahan lainnya yang dijilid menjadi satu pada salah satu ujungnya dan berisi tulisan atau gambar. Setiap sisi dari sebuah lembaran kertas pada buku disebut sebuah halaman (dosenpendidikan, 2020). Buku disusun berdasarkan klasfikasinya, klasfikasi adalah suatu kegiatan mengelompokkan. Dimana klasifikasi sangat dibutuhkan dalam perpusta kaan, karena klasifikasi bertujuan untuk mengelompokkan satu koleksi sejenis, yang pengelompokkannya berdasarkan judul, pengarang, dan lain sebagainya (Lestari, 2016).

Secara umum klasifikasi terbagi ke dalam dua jenis, yaitu :

1. Klasifikasi artifisial, yaitu klasifikasi bahan pustaka berdasarkan sifat-sifat yang kebetulan ada pada bahan pustaka tersebut. Misalnya tinggi buku atau warna buku.
2. Klasifikasi fundamental, yaitu klasifikasi bahan pustaka berdasarkan isi atau subjek buku pada bahan pustaka tersebut (Saputro, 2017).
3. **Algoritma Pencarian (*Search Algorithm*)**

Algoritma adalah metode efektif diekspresikan sebagai rangkaian terbatas dari instruksi-instruksi yang telah didefinisikan dengan baik untuk menghitung sebuah fungsi. Dimulai dari sebuah kondisi awal dan input awal, instruksi-instruksi tersebut

menjelaskan sebuah komputasi yang bila dieksekusi akan diproses lewat sejumlah urutan kondisi terbatas yang terdefinisikan dengan baik, yang pada akhirnya menghasilkan keluaran dan berhenti di kondisi akhir. Transisi dari satu kondisi ke kondisi lainnya tidak harus deterministik. Beberapa algoritma dikenal dengan algoritma pengacakan menggunakan masukan acak.

Pencarian adalah salah satu operasi dasar dari Struktur Data, yang digunakan untuk menemukan data tertentu dari daftar data. Pencarian dapat beroperasi pada daftar elemen yang diurutkan dan tidak diurutkan. Bergantung pada jenis struktur data, algoritma pencarian yang sesuai dipilih (Sultana *dkk.*, 2017). Algoritma pencarian atau *searching algorithm* adalah algoritma yang menerima sebuah argumen kunci dengan langkah-langkah tertentu akan mencari rekaman dengan kunci tersebut. Setelah proses pencarian dilaksanakan, akan diperoleh salah satu dari dua kemungkinan, yaitu data yang dicari ditemukan (*successful*) atau tidak ditemukan (*unsuccessful*) (Abdurrahman, 2016).

1. **Algoritma *Bi-linear Search***

Pencarian *Bi Linear* sesi pencarian baru ini dikemukakan oleh peneliti Smita Paira *dkk.* di tahun 2014. Algoritma bekerja pada daftar elemen yang diurutkan dan tidak diurutkan. Seperti namanya yaitu *Bi Linear*, pencarian menyebar di kedua ujung daftar untuk item pencarian dalam daftar sehingga skenario kasus terburuknya jauh lebih baik daripada algoritma pencarian lainnya (Sultana dkk., 2017).

Efisiensi algoritma pencarian terletak pada kompleksitas waktunya atau jumlah perbandingan yang dibuat. Misalkan jumlah elemen dari sebuah *array* adalah n. Untuk satu iterasi, saat nilai i meningkat, nilai j menurun secara bersamaan. Algoritme hanya membuat satu perbandingan untuk posisi pertama dan terakhir dari larik. Itu membuat dua perbandingan untuk posisi kedua dan kedua hingga terakhir dari larik. Jadi, untuk dua posisi larik hanya dibuat satu perbandingan dan untuk empat posisi larik, dibuat dua perbandingan dan seterusnya. Jadi, jika ada n jumlah elemen, loop iterasi sebanyak n/ 2 kali. Oleh karena itu, kompleksitas waktu kasus rata-rata Pencarian Bi-linear adalah O (n). Jika item ada di posisi pertama dan terakhir dari larik, maka hanya satu perbandingan yang dibuat. Eksekusi pertama dari loop menemukan item tersebut dan proses pencarian berhenti. Oleh karena itu, kompleksitas waktu kasus terbaik dan kasus terburuk dari Pencarian Bi-linear adalah O (1).

Data disimpan di berbagai server seperti server email, server file, dll untuk keamanan. Untuk mengambil data ini, Pencarian *Bi-linear* dapat digunakan. Ini memungkinkan penyisipan, penghapusan, dan segala jenis modifikasi pada posisi yang dicari dalam waktu singkat. Ini dapat digunakan untuk menemukan *bug* yang ada dalam program besar. Pencarian *Bi-linear* dapat digunakan untuk mencari *disk* tertentu dari *drive,* bahkan dalam kasus ruang memori yang sempit. Ia dapat menemukan rekor terakhir dengan hanya membuat satu perbandingan. Ia bahkan tidak memerlukan penyimpanan catatan yang dipesan. Pencarian *Bi-linear* dapat digunakan untuk menemukan nama seseorang di direktori telepon, untuk mencari catatan siswa dari *database* siswa, untuk melacak semua jenis file dari direktori dan banyak lagi dalam waktu singkat. Memiliki waktu eksekusi yang lebih sedikit. Pencarian *Bi-linear* sangat efisien dan lebih cepat dalam menemukan data dari kumpulan data yang besar (Paira dan Chandra, 2014).

### 2.4.1. *Pseudo Code*

Langkah 1 : Masukkan nilai data.

Langkah 2 : Tentukan index.

Langkah 3 : Inisialisasi i = 0 dan j = n-1.

Langkah 4 : Periksa apakah data [i] sama dengan target. Jika ya, lanjutkan ke

langkah 9, jika tidak lanjutkan ke langkah 5.

Langkah 5 : Periksa apakah data [j] sama dengan target. Jika ya, lanjutkan ke langkah 8, jika tidak lanjutkan ke langkah 6.

Langkah 6 : i = i + 1 dan j = j – 1, kembali ke langkah 4.

Langkah 7 : Jika i > (n/2) atau j < (n/2) cetak “Target Tidak Ditemukan”. Lanjutkan ke langkah 10. Jika tidak, Kembali ke langkah 4.

Langkah 8 : Cetak bahwa “Target Ditemukan di Index ke i”, lanjutkan ke langkah 10.

Langkah 9 : Cetak bahwa “Target Ditemukan di Index ke j”, lanjutkan ke langkah 10.

Langkah 10 : Selesai.

### 2.4.2. Proses Perhitungan Metode *Bi-linear Search*

Data :

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 15 | 23 | 41 | 39 | 40 | 16 | 17 | 18 | 56 | 67 |

Jumlah Data (n) : 10

Index :

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |

Batas Bawah (i) : 0

Batas Atas (j) : n – 1

: 10 – 1 = 9

Item Yang Dicari (Target) : 41

Jawab :

* + - 1. Data [i] = Target

Data [0] = 41

15 = 41

15 ≠ 41

* + - 1. Data [j] = Target

Data [9] = 41

67 = 41

67 ≠ 41

1. i = i + 1

i = 0 + 1

i = 1

j = j – 1

j = 9 – 1

j = 8

4. Data [i] = Target

Data [1] = 41

23 = 41

1. ≠ 41
2. Data [j] = Target

Data [8] = 41

56 = 41

1. ≠ 41
2. i = i + 1

i = 1 + 1

i = 2

j = j – 1

j = 8 – 1

j = 7

1. Data [i] = 41

41= 41

1. Selesai.

Jadi, target ditemukan pada index ke 2.

1. ***Hypertext Preprocessor* (PHP)**

*Hypertext Preprocessor***(**PHP) adalah bahasa *scripting* *web* *HTML-embedded*. Ini berarti kode PHP dapat disisipkan ke dalam HTML halaman *Web*. Ketika sebuah halaman PHP diakses, kode PHP dibaca atau"diurai" oleh *server*. *Output* dari fungsi PHP pada halaman biasanya dikembalikan sebagai kode HTML, yang dapat dibaca oleh *browser*. Karena kode PHP diubah menjadi HTML sebelum halaman dibuka, pengguna tidak dapat melihat kode PHP pada halaman. Ini membuat halaman PHP cukup aman untuk mengakses *database* dan informasi aman lainnya. (Ferdianto, 2013).

Banyak sintaks PHP yang hasil adaptasi dari bahasa lain seperti bahasa C, Java dan Perl. Namun, PHP memiliki sejumlah fitur unik dan fungsi tertentu juga. Tujuan dari bahasa pemrograman PHP adalah untuk memungkinkan pengemban *web* untuk menulis halaman yang dihasilkan secara dinamis dengan cepat dan mudah. PHP juga bagus untuk menciptakan situs *Web* *database-driven*. Jika Anda ingin mempelajari lebih lanjut tentang PHP, situs resminya yaitu PHP.net. (Ferdianto, 2013).

Beberapa kelebihan PHP dari bahasa pemrograman *web*, antara lain:

1. Bahasa pemrograman PHP adalah sebuah bahasa *script* yang tidak melakukan sebuah kompilasi dalam penggunaanya.
2. *Web Server* yang mendukung PHP dapat ditemukan dimana-mana dari mulaiapache, IIS, Lighttpd, hingga Xitami dengan konfigurasi yang relatif mudah.
3. Dalam sisi pengembangan lebih mudah, karena banyaknya milis-milis dan *developer* yang siap membantu dalam pengembangan.
4. Dalam sisi pemahamanan, PHP adalah bahasa *scripting* yang paling mudah karena memiliki referensi yang banyak.
5. PHP adalah bahasa *open source* yang dapat digunakan diberbagai mesin (Linux, Unix, Macintosh, Windows) dan dapat dijalankan secara *runtime* melalui *console* serta juga dapat menjalankan perintah-perintah sistem. (Ferdianto, 2013).
6. ***CodeIgniter***

*CodeIgniter* adalah sebuah *framework* PHP yang bersifat *open source* yang menggunakan metode MVC (*model, view,* dan *controller*). *CodeIgniter* bertujuan untuk memudahkan *developer* atau programmer dalam mebangun aplikasi berbasis web (Setiawan Dimas, 2019).

Beberapa keunggulan yang kemudian ditawarkan oleh *CodeIgniter* yaitu sebagai berikut:

1. *CodeIgniter* adalah *framework* yang bersifat *free* dan *open source*.
2. *CodeIgniter* memiliki ukuran yang kecil dibandingkan dengan *framework* lain. Setelah proses instalasi, *framework* *CodeIgniter* hanya berukuran kurang lebih 2 MB. Dokumentasi *CodeIgniter* memiliki ukuran sekitar 6 MB.
3. Aplikasi yang dibuat menggunakan *CodeIgniter* bisa berjalan cepat.
4. *CodeIgniter* menggunakan pola desain *Model-View-Controller* (MVC) sehingga satu file tidak terlalu berisi banyak kode. Hal ini menjadikan kode lebih mudah dibaca, dipahami, dan dipelihara dikemudian hari.
5. *CodeIgniter* dapat diperluas sesuai dengan kebutuhan.
6. *CodeIgniter* terdokumentasi dengan baik. Informasi tentang pustaka kelas dan fungsi yang disediakan oleh *CodeIgniter* dapat diperoleh melalui dokumentasi yang disertakan di dalam paket distribusinya.
7. ***Database Management System***

*Database Management**System* (DBMS) merupakan perangkat lunak untuk mengendalikan pembuatan, pemeliharaan, pengolahan, dan penggunaan data yang berskala besar. Penggunaan DBMS saat ini merupakan hal yang sangat penting dalam segala aspek, baik itu dalam skala yang besar atau kecil. Sebagai contoh media sosial facebook menggunakan DBMS untuk menyimpan data-data pengguna facebook yang sangat banyak kedalam DBMS *MySQL*. Beberapa DBMS yang digunakan adalah *MySQL* dan MariaDB. Berdasarkan survey yang dilakukan, *MySQL* dan MariaDB merupakan DBMS yang banyak digunakan sebagai contoh survey yang terdapat pada db-engines.com *DB-Engines Ranking* menempatkan *MySQL* pada posisi ke-2 sedangkan MariaDB pada posisi ke-20 namun pada *survey* yang terdapat di *serverwatch*.com *Top 10 Enterprise Database Sistem Of 2016*, MariaDB menempati posisi ke-6 dan *MySQL* menempati posisi ke-7 (Wulan Nafesa Septine, S.T., 2019).

1. ***MySQL***

*MySQL* merupakan sebuah perangkat lunak sistem manajemen basis data SQL atau DBMS *Multithread* dan *multi user*. *MySQL* sebenarnya merupakan turunan dari salah satu konsep utama dalam *database* untuk pemilihan atau seleksi dan pemasukan data yang memungkinkan pengoperasian data dikerjakan secara mudah dan otomatis. *MySQL* diciptakan oleh Michael "Monty" Widenius pada tahun 1979, seorang *programmer* komputer asal Swedia yang mengembangkan sebuah sistem *database* sederhana yang dinamakan UNIREG yang menggunakan koneksi *low-level ISAM database engine* dengan *indexing* 1.

1. Kelebihan Dan Kekurangan *MYSQL*

Adapun kelebihan dari *MySQL* dalam penggunaanya dalam *database* yakni sebagai berikut:

1. Gratis sehingga *MySQL* dapat dengan mudah untuk mendapatkannya.
2. *MySQL* stabil dalam pengoprasiannya.
3. *MySQL* mempunyai sistem keamanan yang cukup baik.
4. Sangat mendukung transaksi dan mempunyai banyak dukungan dari komunitas.
5. Sangat *fleksibel* dengan berbagai macam *program* Perkembangan dari *MySQL* sangat cepat.

Adapun kelemahan dari *MySQL* dalam penggunaanya dalam *database* yakni sebagai berikut:

* 1. Sulit untuk diaplikasikan pada intansi atau perusahan dengan *database* yang besar.
  2. *Support* yang kurang.
  3. Tidak populer untuk aplikasi *game* dan *mobile*.

1. **XAMPP**

XAMPP adalah sebuah paket perangkat lunak (*software*) komputer yang sistem penamaannya diambil dari akronim kata Apache, *MySQL* (dulu) / MariaDB (sekarang), PHP, dan *Perl*. Sementara imbuhan huruf “X” yang terdapat pada awal kata berasal dari istilah *cross platform* sebagai simbol bahwa aplikasi ini bisa dijalankan di empat sistem operasi berbeda, seperti *OS Linux, OS Windows, Mac OS,* dan juga Solaris. Sejarah mencatat, *software* XAMPP pertama kali dikembangkan oleh tim proyek bernama *Apache* *Friends* dan sampai saat ini sudah masuk dalam rilis versi 7.3.9yang bisa didapatkan secara gratis dengan label GNU (*General Public License*).

Jika dijabarkan secara gamblang, masing-masing huruf yang ada di dalam nama XAMPP memiliki arti sebagai berikut ini:

1. X artinya *Cross Platform*

Merupakan kode penanda untuk *software cross platform* atau yang bisa berjalan di banyak sistem operasi.

1. A artinya *Apache*

Apache adalah aplikasi web server yang bersifat gratis dan bisa dikembangkan oleh banyak orang (*open source*).

1. M artinya *MySQL* / MariaDB

*MySQL* atau MariaDB merupakan aplikasi *database* server yang dikembangkan oleh orang yang sama. *MySQL* berperan dalam mengolah, mengedit, dan menghapus daftar melalui *database*.

1. PHP

Huruf “P” yang pertama dari akronim kata XAMPP adalah inisial untuk menunjukkan eksistensi bahasa pemrograman PHP. Bahasa pemrograman ini biasanya digunakan untuk membuat *website* dinamis, contohnya dalam *website* berbasis CMS WordPress.

1. P artinya *Perl*

Sementara itu, untuk huruf P selanjutnya merupakan singkatan dari bahasa pemrograman *Perl* yang kerap digunakan untuk memenuhi berbagai macam kebutuhan. *Perl* ini bisa berjalan di dalam banyak sistem operasi sehingga sangat fleksibel dan banyak digunakan.

Program aplikasi XAMPP berfungsi sebagai *server* lokal untuk mengampu berbagai jenis data *website* yang sedang dalam proses pengembangan. Dalam prakteknya, XAMPP bisa digunakan untuk menguji kinerja fitur ataupun menampilkan konten yang ada didalam *website* kepada orang lain tanpa harus terkoneksi dengan internet, atau istilahnya *website offline*. XAMPP bekerja secara *offline* layaknya *web hosting* biasa namun tidak bisa diakses oleh banyak orang. Maka dari itu, XAMPP biasanya banyak digunakan oleh para Mahasiswa maupun pelajar untuk melihat hasil desain *website* sebelum akhirnya dibuat *online* menggunakan *web hosting* yang biasa dijual dipasaran.

1. ***Flowchart***

*Flowchart* adalah simbol-simbol pekerjaan yang menunjukkan bagan aliran proses yang saling terhubung sehingga setiap simbol *flowchart* melambangkan pekerjaan dan instruksinya (Heri, 2017). Simbol–simbol yang digunakan dalam *flowchart* adalah sebagai berikut:

**Tabel 2. 1 Simbol-Simbol Flowchart**

|  |  |  |  |
| --- | --- | --- | --- |
| **NO.** | **GAMBAR** | **NAMA** | **KETERANGAN** |
| 1. |  | Proses | Mempresentasikan operasi. |
| 2. |  | *Input / Output* | Mempresentasikan *Input* atau *Output* data yang diproses atau informasi. |
| 3. |  | Keputusan | Keputusan dalam program. |
| 4. |  | Dokumen | Dokument I / O dalam format cetak. |
| 5. |  | *Terminal points* | Awal / akhir *flowchart*. |
| 6. |  | *Preparation* | Pemberian harga awal. |
| 7. |  | Manual *input* | *Input* yang dimasukkan secara manual dari *keyboard*. |
| 8. |  | Penghubung | Keluar atau masuk dari bagian lain *flowchart* khususnya. |
| 9. |  | Penghubung | Keluar atau masuknya dari bagian lain *flowchart* khususnya halaman lain. |
| 10. |  | *Display* | *Output* yang ditampilkan pada terminal. |
| 11. |  | Anak panah | Mempresentasikan alur kerja. |

Sumber: (Heri, 2017).

**2.11.** **Metode Perancangan *Unified Modeling Language* (UML)**

*Unified Modeling Language* (UML) adalah bahasa permodelan untuk merancang sistem berbasis *object- oriented.* UML digunakan untuk pengembangan sebuah sistem yang dapat menyampaikan alur kerja sistem dan menjelaskan tugas setiap *user* dalam sebuah sistem. UML memiliki tiga kategori diagram, yaitu *strucrture diagrams, behavior diagram,* dan *interaction diagram.* Ada beberapa jenis diagram dalam UML:

1. *Use Case Diagram*

*Use case diagram* atau diagram *use case* merupakan permodelan untuk kelakuan (*behavior*) sistem informasi yang akan dibuat. *Use case* mendeskripsikan sebuah interaksi antara satu atau lebih *actor* dengan sistem informasi yang akan dibuat. Secara kasar, *use case* digunakan untuk mengetahui fungsi apa saja yang ada di dalam sebuah sistem informasi dan siapa saja yang berhak menggunakan fungsi- fungsi itu (Kiswanto, 2016).

**Tabel 2. 2 Simbol-Simbol Use Case**

|  |  |  |  |
| --- | --- | --- | --- |
| **No** | **Simbol** | **Nama** | **Keterangan** |
| 1. |  | *Use Case* | Fungsionalitasi yang disediakan sistem sebagai unit-unit yang saling bertukar pesan antar unit atau *actor*; biasanya dinyatakan dengan menggunakan kata kerja diawal frase nama *use case.* |

**Tabel 2. 2Simbol-Simbol Use Case (lanjutan)**

|  |  |  |  |
| --- | --- | --- | --- |
| **No** | **Simbol** | **Nama** | **Keterangan** |
| 2. |  | Aktor */ actor* | Orang proses, atau sistem lain yang berinteraksi dengan sistem informasi yang akan dibuat diluar sistem informasi yang akan dibuat itu sendiri. |
| 3. |  | *Asosiasi / association* | Komuniksi antara aktor dan *use case* yang berpartisipasi pada *use case* atau *use case* memiliki interaksi dengan *actor*. |
| 4. | << extend >> | *Ekstensi / extend* | *Relasi* *use case* tambahan ke sebuah *use case* dimana *usecase* yang Ditambahkan dapat berdiri sendiri walau tanpa use case tambahan. |
| 5. |  | *Generalisasi*  */generalizati on* | Hubungan generalisasi dan spesialiasi (umum - khusus) antara dua buah use case dimana fungsi yang satu adalah fungsi yang lebih umum dari lainnya. |

Sumber: (Kiswanto, 2016).

1. *Activity Diagram*

*Activity diagram* atau diagram aktivitas menggambarkan *workflow* (aliran kerja) atau aktivitas dari sebuah sistem atau proses bisnis atau *menu* yang ada pada perangkat lunak (Kiswanto, 2016). Yang perlu diperhatikan adalah bahwa diagram aktivitas menggambarkan aktivitas sistem bukan apa yang dilakukan aktor.

**Tabel 2. 3 Simbol-Simbol Activity Diagram**

|  |  |  |  |
| --- | --- | --- | --- |
| **No** | **Simbol** | **Nama** | **Keterangan** |
| 1. |  | *Start* / Status awal | Status awal aktivitas sistem, sebuah diagram aktivitas memiliki sebuah status awal |
| 2. |  | Aktivitas | Aktivitas yang dilakukan sistem, aktivitas biasanya diawali dengan kata kerja |
| 3. |  | Percabangan / *desicion* | Asosiasi percabangan dimana jika ada pilihan aktivitas lebih dari satu |
| 4. |  | Penggabungan / *join* | Asosiasi penggabungan dimana lebih dari satu aktivitas digabungkan menjadi satu. |
| 5. |  | Status akhir | Status akhir yang dilakukan sistem, sebuah diagram aktivitas memiliki sebuah status akhir. |
| 6 |  | *swimlane* | Memisahkan organisasi bisnis yang bertanggung jawab terhadap aktivitas yang terjadi |

Sumber: (Kiswanto, 2016).

1. *Class Diagram*

*Class diagram* atau diagram kelas menggambarkan struktur sistem dari segi pendefinisian kelas-kelas yang akan dibuat untuk membangun sistem. Diagram kelas dibuat agar *programmer* dapat membuat kelas-kelas didalam *program* perangkat lunak sesuai dengan perancangan diagram kelas (Kiswanto, 2016). Ada beberapa simbol-simbol yang terdapat dalam diagram kelas, yaitu:

**Tabel 2. 4 Simbol-Simbol Class Diagram**

|  |  |  |  |
| --- | --- | --- | --- |
| **No** | **Simbol** | **Nama** | **Keterangan** |
|  | |  | | --- | | **nama\_kelas** | | + atribut | | + operasi() | | Kelas | Kelas pada struktur sistem |
| 2. |  | Antarmuka / *interface* | Semua dengan konsep *interface* dalam  pemrograman b*error* ientasi objek |
| 3 |  | Asosiasi / *association* | Relasi antarkelas dengan makna umum, asoasi biasanya juga disertai |
| 4 |  | Asosiasi berarah  /*directed asosiasi* | Relasi antarkelas dengan makna kelas yang satu  digunakan oleh kelas yang lain, asosiasi biasanya juga disertai dengan *multiplicity* |
| 5 |  | Generalisasi | Relasi antarmuka dengan makna generalisasi spesialisasi (umum khusus) |
| 6 |  | Kebergantungan / *dependency* | Relasi antarmuka dengan makna kebergantungan antarkelas |

Sumber: (Kiswanto, 2016).

1. *Sequence Diagram*

*Sequence diagram* menggambarkan kelakuan objek pada *use case* dengan mendeskripsikan waktu hidup objek dan *message* yang dikirimkan dan terima antar objek. Oleh karena itu untuk menggambar diagram *sequence* maka harus diketahui objek-objek yang dimiliki kelas yang diinstasiasi menjadi objek itu. Menggambar diagram  *sequence* juga dibutuhkan untuk melihat *scenario* yang ada pada *use case* (Kiswanto, 2016).

**Tabel 2. 5 Simbol-Simbol Sequence Diagram**

|  |  |  |  |
| --- | --- | --- | --- |
| **No** | **Simbol** | **Nama** | **Keterangan** |
| 1. |  | Aktor | Orang proses, atau sistem lain yang berinteraksi dengan sistem informasi yang akan dibuat diluar sistem informasi yang akan dibuat itu sendiri. |
| 2. |  | Garis hidup / *lifeline* | Menyatakan kehidupan suatu objek |
| 3. | Nama objek : nama kelas | Objek | Menyatakan objek yang berinteraksi pesan |
| 4. |  | Waktu aktif | Menyatakan objek dalam keadaan aktif dan berinteraksi, semua yang terhubung dengan waktu aktif ini adalah sebuah tahapan yang dilakukan didalamnya. |
| 5. |  | Pesan tipe *create* | Menyatakan suatu objek membuat objek yang lain, arah panah mengarah pada objek yang dibuat |

**Tabel 2.5 Simbol-Simbol Sequence Diagram (lanjutan)**

|  |  |  |  |
| --- | --- | --- | --- |
| **No** | **Simbol** | **Nama** | **Keterangan** |
| **7** |  | Pesan tipe *send* | Menyatakan bahwa suatu objek mengirim data / masukkan / informasi ke objek lainnya, arah panah mengarah pada objek yang dikirimi |
| 8. |  | Pesan tipe *return* | Menyatakan bahwa suatu objek yang telah menjalankan suatu operasi atau metode menghasilkan suatu kembalian ke objek tertentu, arah panah mengarah pada objek yang menerima kembalian |
| 9. |  | Pesan *type* *distroy* | Menyatakan suatu objek mengakhiri hidup objek yang lain, arah panah mengarah pada objek yang diakhiri,  sebaiknya jika ada create makan ada *destroy* |

Sumber: (Kiswanto, 2016).

# **BAB III METODOLOGI PENELITIAN**

1. **Waktu dan Tempat Penelitian**
   * 1. **Waktu**

Waktu pelaksaan penelitian tugas akhir dilaksanakan mulai dari Bulan Juli 2021 sampai dengan September 2021. Rincian Kegiatan dapat dilihat pada Tabel 3.1 berikut:

**Tabel 3. 1 Gannt Chart Waktu Penelitian**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **No** | **Uraian** | **Waktu (2021)** | | | | | | | | | | | |
| **Juli** | | | | **Agustus** | | | | **September** | | | |
| **1** | **2** | **3** | **4** | **1** | **2** | **3** | **4** | **1** | **2** | **3** | **4** |
| **1** | ***Inception*** |  |  |  |  |  |  |  |  |  |  |  |  |
| **2** | ***Elaboration*** |  |  |  |  |  |  |  |  |  |  |  |  |
| **3** | ***Construction*** |  |  |  |  |  |  |  |  |  |  |  |  |
| **4** | ***Transition*** |  |  |  |  |  |  |  |  |  |  |  |  |

* + 1. **Tempat Penelitian**

Adapun tempat penelitian tugas akhir yaitu akan dilakukan di Perpustakaan SMA Negeri 5 Kendari.

1. **Metode Pengumpulan Data**

Dalam penelitian ini metode pengumpulan data yang digunakan pada perancangan aplikasi tersebut adalah studi literatur. Metode ini dilaksanakan dengan melakukan studi kepustakaan yang relevan. Metode ini dilakukan untuk mencari sumber pelengkap yang berhubungan dengan aplikasi yang akan dibangun, yaitu dengan mencari referensi yang berkaitan dengan tugas akhir, mulai dari buku-buku, jurnal maupun artikel dan sumber-sumber lain di *internet*. Serta mengumpulkan

dokumen-dokumen yang berkaitan dengan metode *Bi-Linear Search*yang akan diterapkan pada penelitian ini.

1. **Metode Pengembangan Sistem**

Metode pengembangan sistem yang digunakan dalam sistem ini adalah metode *Rational Unified Process* (RUP). Dalam metode ini, terdapat empat tahap pengembangan perangkat lunak, yaitu:

* + 1. **Permulaan (*Inception*)**

Pada fase ini dilakukan proses pengidentifikasian sistem, dilakukan dengan analisis kebutuhan akan aplikasi, melakukan kajian terhadap penelitian yang berkaitan dengan algoritma *bi-linear search*.

* + 1. **Perluasan/Perencanaan (*Elaboration*)**

Setelah menentukan ruang lingkup penelitian, tahap ini akan dilakukan perancangan dan analisis sistem menggunakan *flowchart*,yaitu *flowchart* sistem dan *flowchart* algoritma *bi-linear search.* Pada perancangan ini, digunakan juga *UML* (*Unified Modelling Language*) yang meliputi *use case diagram,* *activity diagram, class diagram* dan *sequence diagram* dari perangkat lunak yang akan dibuat, serta desain antarmuka aplikasi yang akan dibuat.

* + 1. **Konstruksi (*Construction*)**

Proses yang dilakukan pada tahap ini yaitu membangun aplikasi dengan perancangan yang telah dilakukan sebelumnya, mulai dari tampilan *interface* sampai implementasi rancangan UML. Proses yang juga dilakukan pada tahap ini yaitu penerapan *coding* algoritma *Bi-Linear Search*sebagai metode pencarian.

* + 1. **Transisi (*Transition*)**

Pada tahap ini difokuskan untuk melakukan pengujian terhadap sistem pencarian. Memperbaiki masalah yang muncul saat pembuatan dan setelah pengujian aplikasi.

1. **Analisis Sistem**

Analisis sistem merupakan suatu tahapan yang bertujuan untuk mengetahui dan mengamati semua yang terlibat dalam suatu sistem. Pembahasan yang ada pada analisis sistem ini yaitu analisis kebutuhan fungsional meliputi perancangan sistem menggunakan bahasa pemodelan UML (*Unified Modeling Language*), perancangan tampilan *interface* serta analisis kebutuhan fungsional dan non fungsional yang meliputi kebutuhan perangkat keras dan perangkat lunak yang akan digunakan.

### 3.4.1. Analisis Kebutuhan Fungsional

Analisis kebutuhan fungsional adalah segala bentuk data yang dibutuhkan oleh sistem agar sistem dapat berjalan sesuai dengan prosedur yang dibangun melalui perancangan sistem. Adapun kebutuhan fungsional dari sistem yang akan dibangun yaitu Perancangan diagram sistem menggunakan bahasa pemodelan UML (*Unified Modeling Language*) yang meliputi *flowchart* sistem, *flowchart* metode, *use case diagram*, *activity diagram*, *class diagram*, serta *sequence diagram* dan Desain *Interface* sistem.

### 3.4.2. Analisis Kebutuhan Non Fungsional

Analisis kebutuhan non fungsional adalah sebuah langkah dimana pembangun aplikasi menganalisis sumber daya kebutuhan untuk membangun aplikasi yang akan dibangun. Analisis kebutuhan nonfungsional yang dilakukan dibagi dalam dua tahap, yaitu anlisis kebutuhan perangkat keras dan analisis kebutuhan perangkat lunak. Kebutuhan perangkat keras yaitu kebutuhan perangkat atau komponen yang dibutuhkan pada sistem dan perangkat lunak yaitu kebutuhan perangkat lunak untuk membantu agar komponen perangkat keras dapat berfungsi dan dapat dijalankan pada sistem.

#### 3.4.2.1. Kebutuhan Perangkat Keras

Untuk menerapkan rancangan yang telah dijelaskan sebelumnya, dibutuhkan beberapa perangkat keras sebagai sarana untuk mengimplementasikan aplikasi yang dibangun. Berikut ini spesifikasi perangkat keras yang dibutuhkan.

**Tabel 3. 2 Spesifikasi Perangkat Keras**

|  |  |  |
| --- | --- | --- |
| No | Nama Perangkat | Spesifikasi |
| 1. | *PC* | *Toshiba l-40* |
| 2. | *Processor* | *Intel dual core* |

**Tabel 3. 3 Spesifikasi Perangkat Keras (lanjutan)**

|  |  |  |
| --- | --- | --- |
| No | Nama Perangkat | Spesifikasi |
| 3. | *RAM* | *4 GB* |
| 4. | *Harddisk* | *128 GB SSD* |
| 5. | *Monitor* | *Monitor 14 Inch* |

#### 3.4.2.2. Kebutuhan Perangkat Lunak

Perangkat lunak yang digunakan pada pembangunan *system* ini. Adapun rincian kebutuhan perangkat lunak dapat dilihat pada Tabel 3.3 berikut ini.

**Tabel 3. 4 Spesifikasi Perangkat Lunak**

|  |  |  |  |
| --- | --- | --- | --- |
| No. | Nama Perangkat | Fungsi | Spesifikasi |
| 1. | *Windows* | *Operating System* | *Windows 10 Home* |
| 2. | *Xampp* | Universal development environment | *Xampp v3.2.2* |
| 3. | *MySQL* | *Database Management* | *MySQL 5.7* |
| 4. | *PHP* | *Web development* | *PHP 7.3* |
| 5. | *Visual Studio Code* | *Code Editor* | *VS Code v1.38* |
| 6. | *Boostrap* | *Framework CSS* | *Version 1.41* |
| 7. | *CodeIgniter* | *Framework PHP* | *Version 3.1.11* |

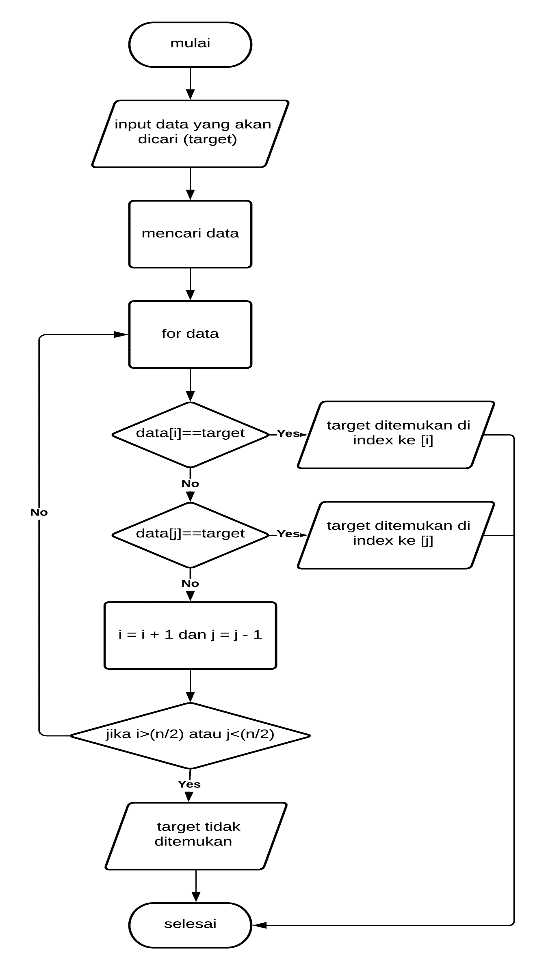
1. **Analisis Perancangan Sistem**

Perancangan sistem yang akan dibangun terdiri atas perancangan *flowchar*t, perancangan UML serta perancangan *user interface.*

### 3.5.1. Perancangan *Flowchart*

*Flowchart* adalah simbol-simbol pekerjaan yang menunjukkan bagan aliran proses yang saling terhubung. Perancangan *flowchart* yang akan dibangun terdiri atas *flowchart system.* Dan adapun alur kerja *Flowchart System* adalah sebagai berikut.

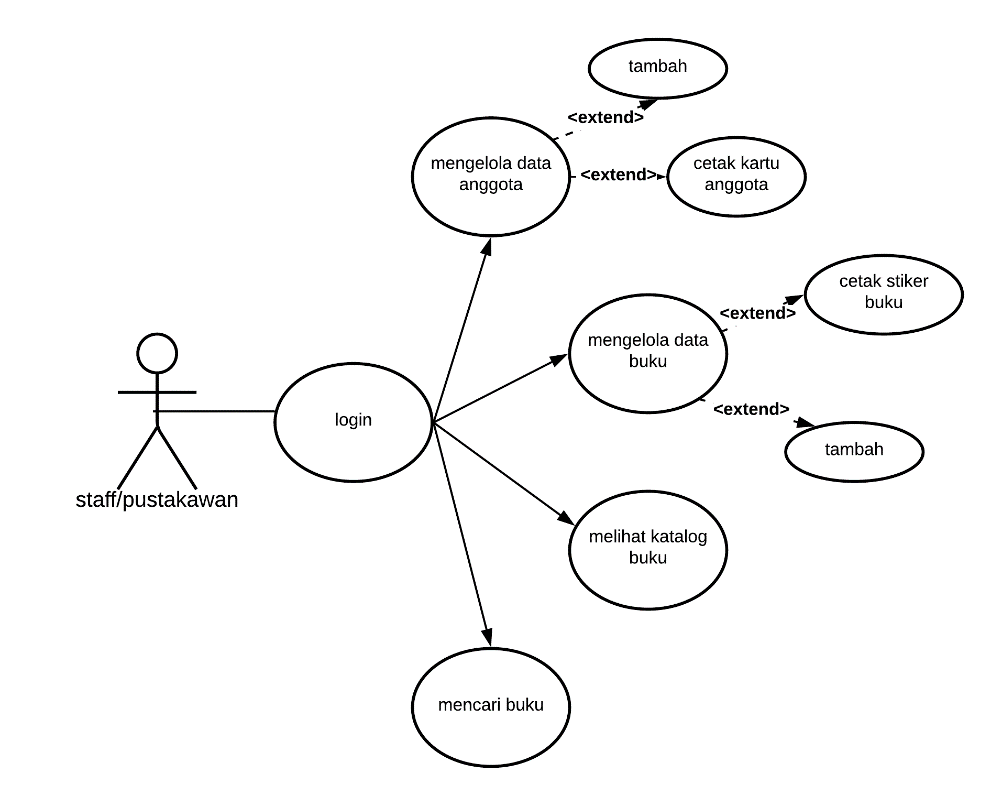
1. User menginput data yang akan dicari (target)
2. *System* mencari data.
3. Jika data [i] sama dengan target yang dicari maka proses selesai. Jika tidak, maka lanjut periksa data [j] apakah sama dengan target dengan data yang dicari.
4. Jika data belum ditemukan di index [i] dan [j], maka [i] diikremen dan [j] didikremen.
5. Jika i > (n/2) atau j < (n/2) berarti data tidak ditemukan. Jika tidak, maka proses pencarian data diulang.
6. Proses selesai.



**Gambar 3. 1 Flowchart System**

### 3.5.2. *Use Case Diagram*

Use case diagram digunakan untuk memodelkan fungsionalitas-fungsionalitas system yangn dilihat dari pengguna yang ada di luar *system* (*actor*). Berikut adalah use case diagram system yang dibangun pada Gambar 3.2.



**Gambar 3. 2 Use Case Diagram**

1. Deskripsi *Use* *Case Login*

**Tabel 3.4 Deskripsi *Use* *Case Login***

|  |  |
| --- | --- |
| **Nama** | ***Manage*** |
| Deskripsi Singkat | *Actor* dapat *login* untuk masuk ke sistem informasi perpustakaan SMA Negeri 5 Kendari. |
| Kondisi Awal | Halaman *login* terbuka dan *actor* harus memasukkan  *username* dan *password*. |
| Kondisi Akhir | *Actor* telah terverifikasi sebagai pengguna *system* informasi perpustakaan. |
| Situasi Kesalahan | Ketika *username* atau *password* yang dimasukkan  *actor* salah. |

**Tabel 3.4 Deskripsi *Use* *Case Login* (lanjutan)**

|  |  |
| --- | --- |
| **Nama** | ***Manage*** |
| Status Sistem Saat terjadi Kesalahan | *Actor* akan selalu dialihkan ke halaman *login* dan  pesan kesalahan. |
| Aktor | *Staff*/Pustakawan |
| *Trigger* | *Actor* harus *login* terlebih dahulu sebelum dapat  masuk ke sistem dan melakukan kegiatan. |
|  |  |
| Deskripsi Singkat | *Actor* dapat *login* untuk masuk ke sistem informasi perpustakaan daerah Kota Kendari Sulawesi Tenggara. |
| Proses standar | 1. *Actor* memasukkan *username* dan *password* 2. Sistem melakukan validasi 3. Sistem menampilkan halaman *home* |
| Proses alternatif | Jika validasi gagal maka sistem akan kembali menampilkan halaman *login*. Berikut pesan kesalahan “Gagal Login, username atau password salah” |

1. Deskripsi *Use* *Case* Mengelolah Data Anggota

**Tabel 3.5 Deskripsi *Use* *Case* Mengelolah Data Anggota**

|  |  |
| --- | --- |
| **Nama** | ***Manage*** |
| Deskripsi Singkat | *Actor* dapat mengelolah data anggota |
| Kondisi Awal | Menu pendaftaran anggota telah terbuka. |
| Situasi Kesalahan | - |
| Status Sistem Saat terjadi Kesalahan | - |
| Aktor | *Staff*/Pustakawan |

**Tabel 3.5 Deskripsi *Use* *Case* Mengelolah Data Anggota (lanjutan)**

|  |  |
| --- | --- |
| **Nama** | ***Manage*** |
| *Trigger* | *Actor* membutuhkan data anggota untuk dapat mengelolah data anggota. |
| Proses Standar | 1. *Actor* masuk mengelolah data anggota. 2. *Actor* mengelolah *button* tambah dan cetak kartu anggota |

1. Deskripsi *Use* *Case* Mengelolah Data Buku

**Tabel 3.6 Deskripsi *Use* *Case* Mengelolah Data Buku**

|  |  |
| --- | --- |
| **Nama** | ***Manage*** |
| Deskripsi Singkat | *Actor* dapat mengelolah data buku. |
| Kondisi Awal | Menu pendaftaran buku telah terbuka. |
| Situasi Kesalahan | - |
| Status Sistem Saat terjadi Kesalahan | - |
| Aktor | *Staff*/Pustakawan |
| *Trigger* | *Actor* membutuhkan data buku untuk dapat mengelolah data buku. |
| Proses Standar | 1. *Actor* masuk mengelolah data buku. 2. *Actor* mengelolah *button* tambah dan cetak stiker buku |

1. Deskripsi *Use* *Case* Melihat Katalog Buku

**Tabel 3.7 Deskripsi *Use* *Case* Melihat Katalog Buku**

|  |  |
| --- | --- |
| **Nama** | ***Manage*** |
| Deskripsi Singkat | *Actor* dapat melihat katalog buku. |
| Kondisi Awal | Menu katalog buku telah terbuka. |
| Situasi Kesalahan | - |

**Tabel 3.7 Deskripsi *Use* *Case* Melihat Katalog Buku (lanjutan)**

|  |  |
| --- | --- |
| **Nama** | ***Manage*** |
| Status Sistem Saat terjadi Kesalahan | - |
| Aktor | *Staff*/Pustakawan |
| *Trigger* | - |
| Proses Standar | 1. *Actor* masuk melihat katalog buku. 2. *Actor* menampilkan tabel katalog buku |

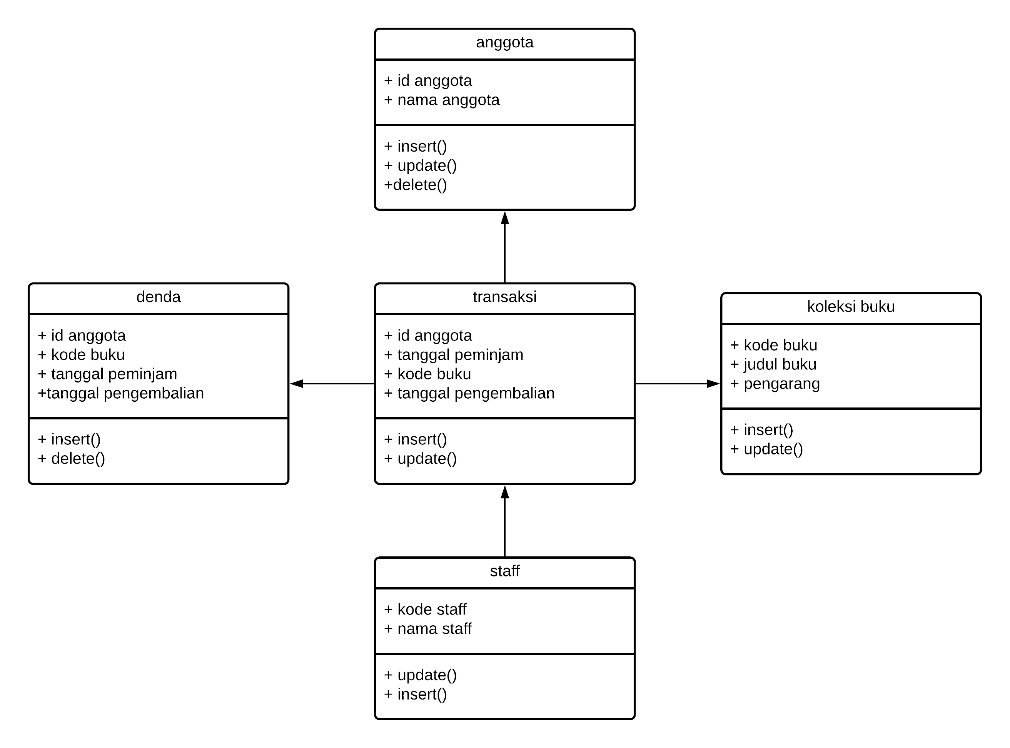
1. Deskripsi *Use* *Case* Mengelolah Data

**Tabel 3.8 Deskripsi *Use* *Case* Mencari Buku**

|  |  |
| --- | --- |
| **Nama** | ***Manage*** |
| Deskripsi Singkat | *Actor* dapat mencari buku. |
| Kondisi Awal | Menu katalog buku telah dibuka |
| Situasi Kesalahan | *Actor* salah menginput *pattern.* |
| Status Sistem Saat terjadi Kesalahan | *System* menampilkan pesan “Tidak Ada Hasil”. |
| Aktor | *Staff*/Pustakawan |
| *Trigger* | *Actor* membutuhkan data katalog buku untuk mencari buku. |
| Proses Standar | 1. *Actor* mencari buku. 2. *System* akan menampilkan daftar katalog buku yang dicari sesuai *data* yang diinput. |

### *3.5.3 Class Diagram*

Pada sistem ini memiliki 5 *class* yaitu *class* anggota, *class* denda, *class* koleksi buku, *class* transaksi, *class staff*. Dimana *class* transaksi memiliki hubungan antar *class* dengan *class* anggota, *class* denda, *class* koleksi buku dan *class staff*.



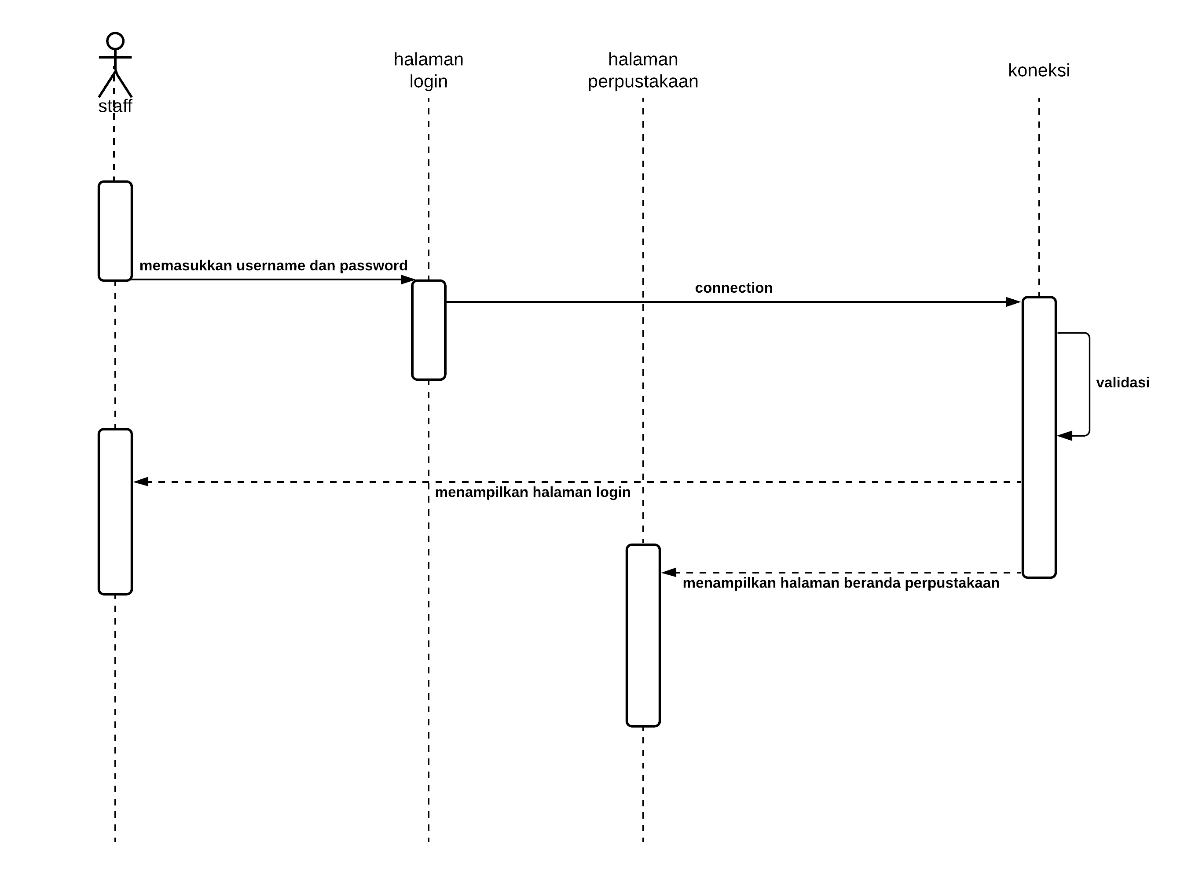
**Gambar 3.3 *Class Diagram***

### 3.5.4 *Sequence Diagram*

*Sequence diagram* adalah diagram yang menggambarkan kolaborasi dinamis antara sejumlah *object*. Kegunaannya untuk menunjukkan rangkaian pesan yang dikirim antara *object* juga interaksi antara *object*. Sesuatu yang terjadi pada titik tertentu dalam eksekusi sistem.

1. *Sequence Diagram Login*

Setelah melakukan analisa sistem, maka didapatkan rancangan *Sequence diagram login* yang ditunjukkan pada Gambar 3.4.

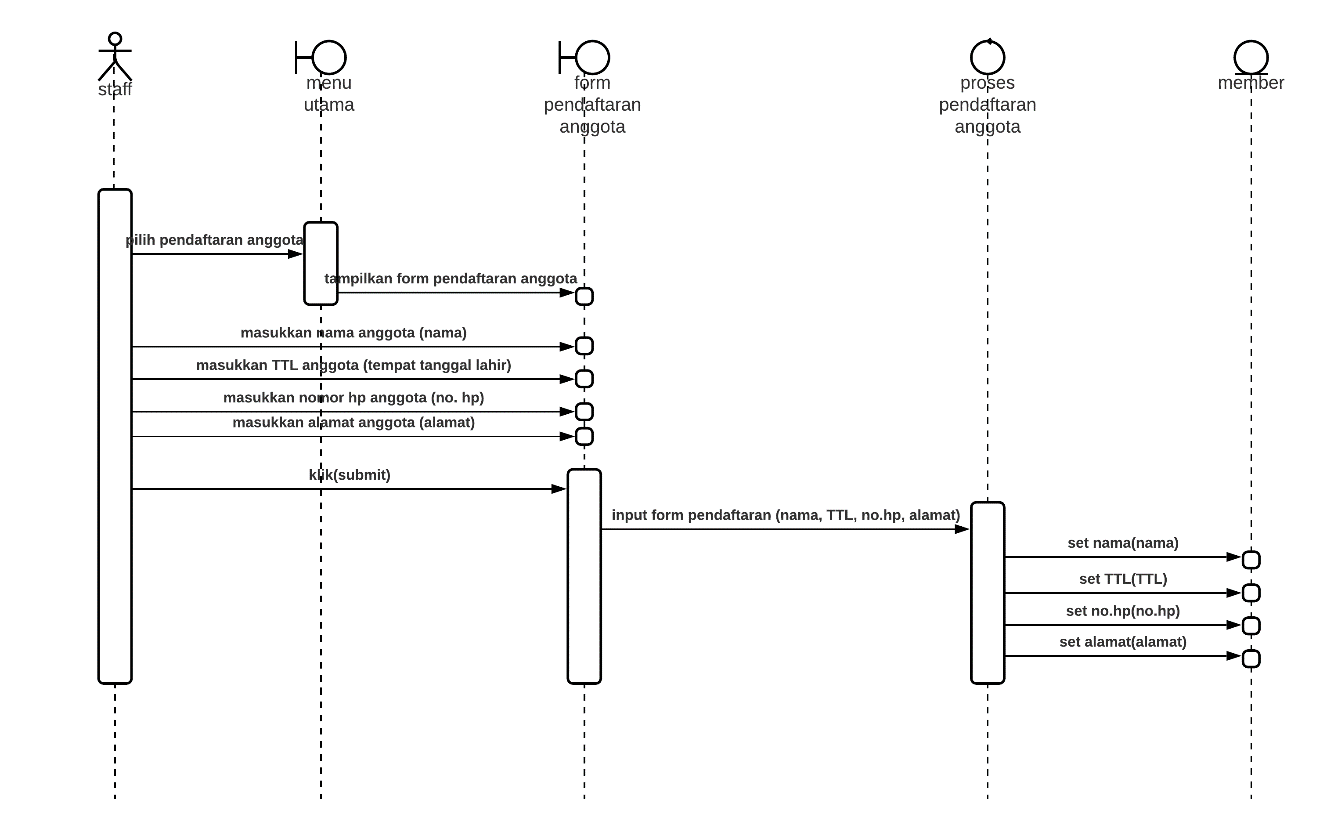
**

**Gambar 3.4 *Sequence Diagram Login***

Adapun alur kerja dari *Sequence diagram login* adalah sebagai berikut :

1. *User* memasukkan *username* dan *password* pada *form login*.
2. Sistem menghubungkan *username* dan *password* ke koneksi, kemudian divalidasi.
3. Jika *username* dan *password* sesuai maka *login* berhasil dan *user* dapat masuk ke halaman perpustakaan. Jika *username* dan *password* salah maka akan kembali ke *user* untuk memasukkan kembali *username* dan *password*.
4. *Sequence Diagram Menambah Anggota*

Setelah melakukan analisa sistem, maka didapatkan rancangan *Sequence diagram* menambah anggota yang ditunjukkan pada Gambar 3.5.

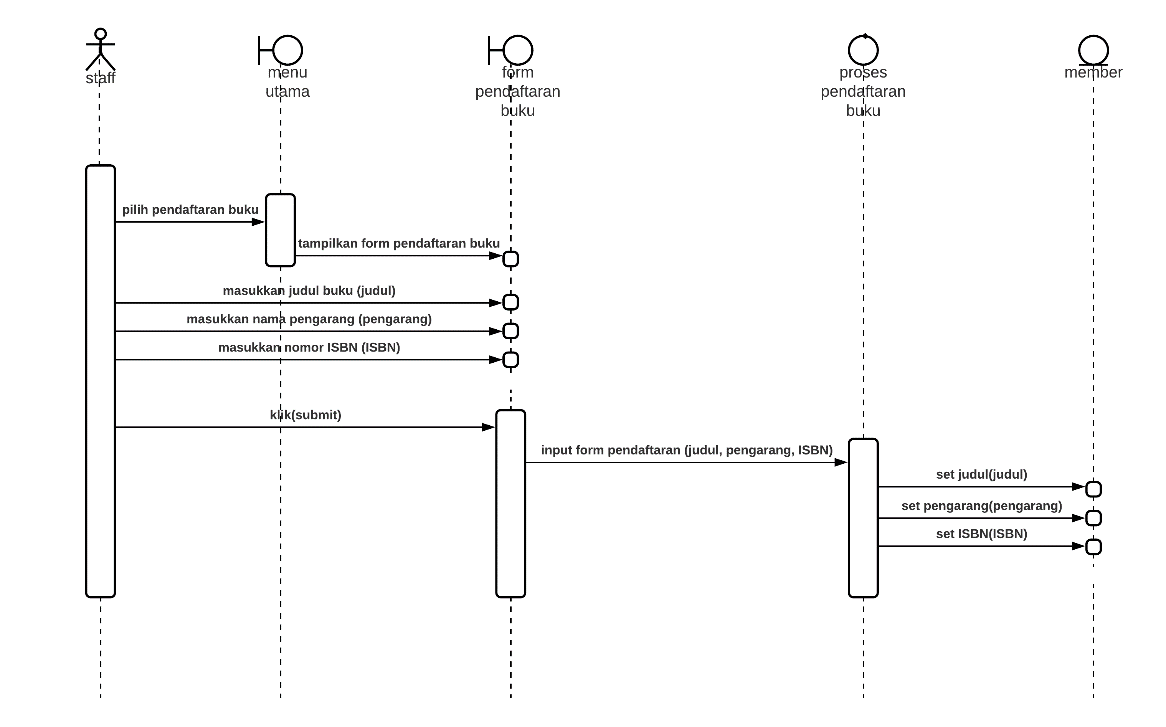
**

**Gambar 3.5 *Sequence* Diagram Menambah Anggota**

Adapun alur kerja dari *Sequence Diagram* menambah anggota adalah sebagai berikut:

1. *User* memilih menu pendaftaran anggota, kemudian *system* menampilkan *form* pendaftaran anggota.
2. *User* mengisi nama, TTL, nomor *handphone*, serta alamat dan mengklik *submit*. Setelah *user* meng-*input* *form* pendaftaran, maka *system* akan memproses data tersebut.
3. *Sequence Diagram* Menambah Buku

Setelah melakukan analisa sistem, maka didapatkan rancangan *Sequence diagram* menambah buku yang ditunjukkan pada Gambar 3.6

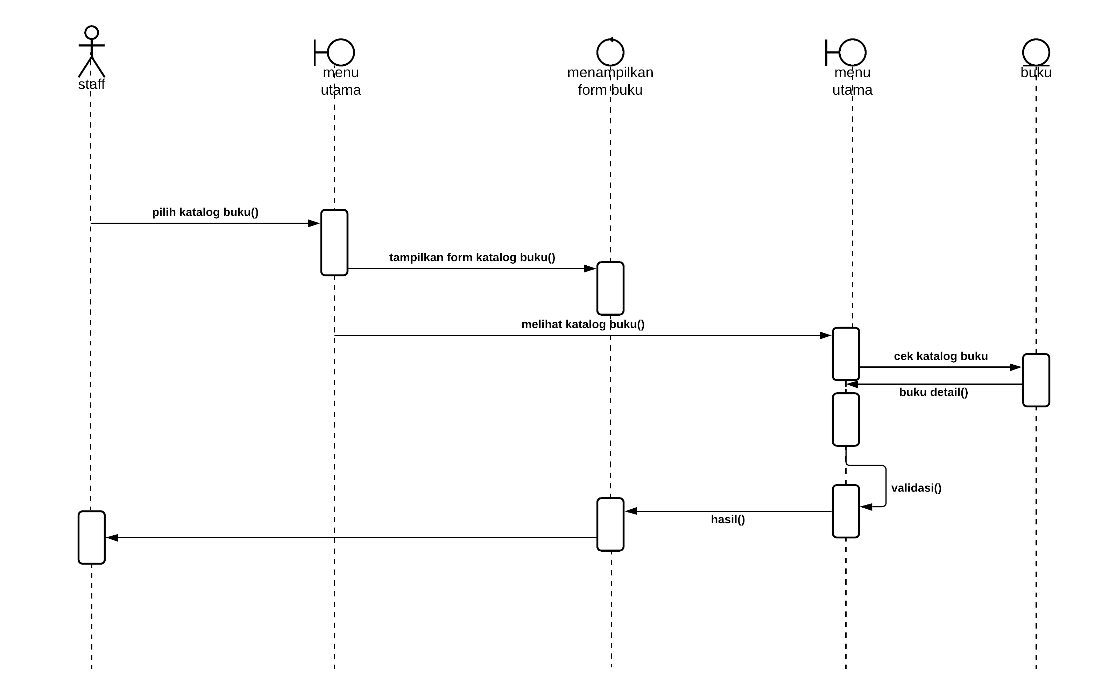
**

**Gambar 3.6 *Sequence* Diagram Menambah Buku**

Adapun alur kerja dari *Sequence Diagram* menambah buku adalah sebagai berikut:

1. *User* memilih menu menambah buku, kemudian *system* menampilkan *form* menambah buku.
2. *User* memasukan judul buku, pengarang, ISBN dan mengklik *submit*. Setelah *user* menginput *form* menambah buku, maka *system* akan memproses data tersebut.
3. *Sequence Diagram* Melihat Katalog

Setelah melakukan analisa sistem, maka didapatkan rancangan *Sequence diagram* melihat katalog yang ditunjukkan pada Gambar 3.7.

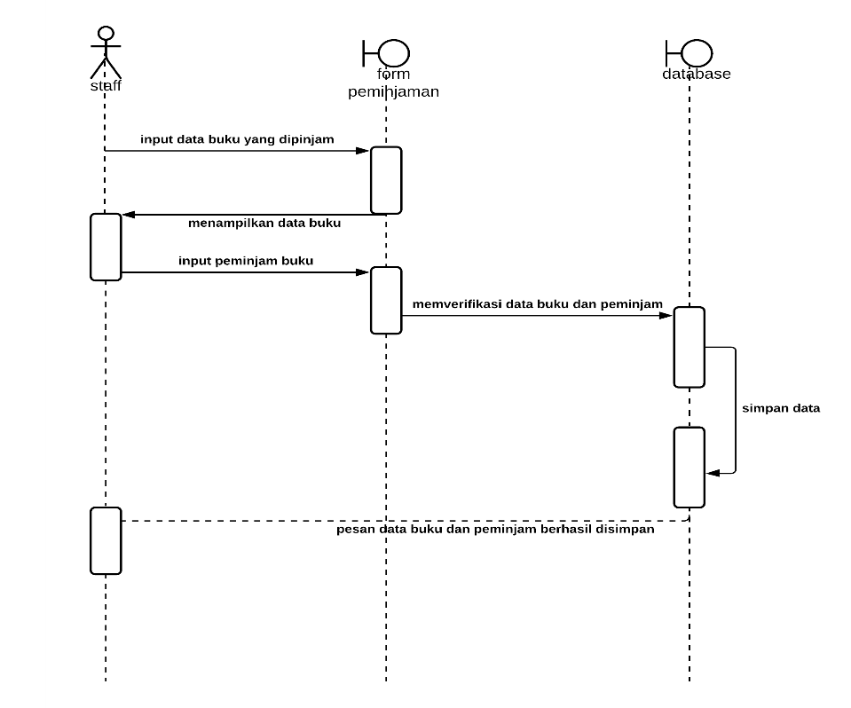


**Gambar 3.7 *Sequence Diagram* Melihat Katalog**

Adapun alur kerja dari *Sequence Diagram* melihat katalog adalah sebagai berikut:

1. *User* memilih menu katalog buku, kemudian *system menampilkan form* melihat katalog
2. *User* melihat katalog buku.
3. *Sequence Diagram* Melihat Katalog

Setelah melakukan analisa sistem, maka didapatkan rancangan *Sequence diagram* meminjam buku yang ditunjukkan pada Gambar 3.8.

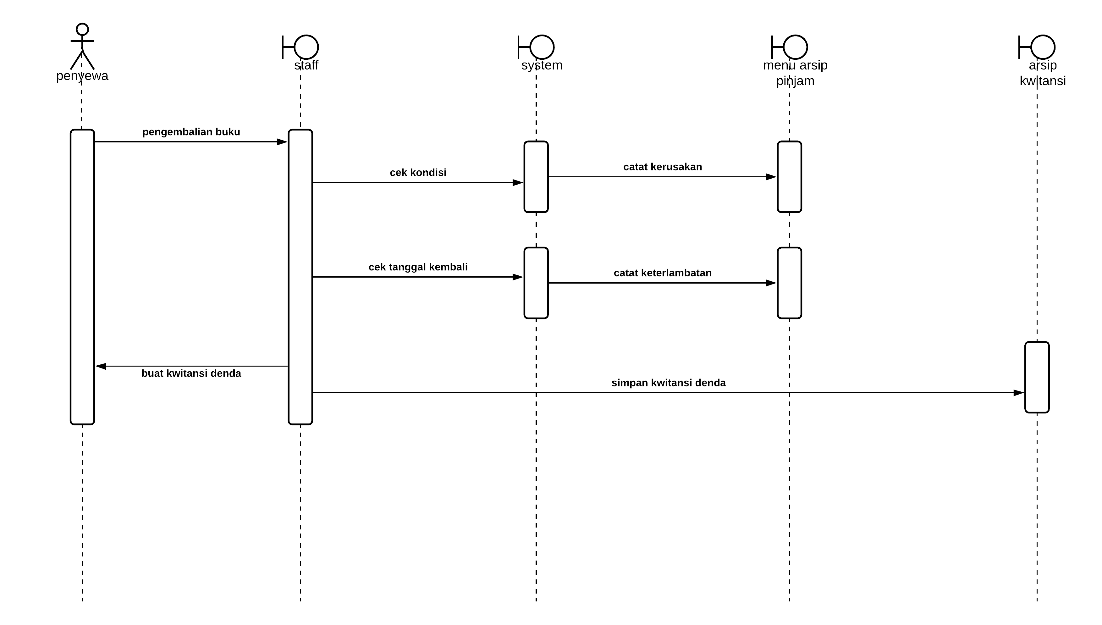
**

**Gambar 3.8 *Sequence* Diagram Meminjam Buku**

Adapun alur kerja dari *sequence* diagram meminjam buku adalah sebagai berikut:

1. *User* menginput data buku yang dipinjam, kemudian *system* menampilkan data buku.
2. *User* menginput peminjam buku, kemudian *system* memverifikasi data buku dan peminjam. *Database* lalu menyimpan data.
3. *System* menampilkan pesan “data buku dan peminjam berhasil disimpan”.
4. *Sequence Diagram* Pengembalian Buku

Setelah melakukan analisa sistem, maka didapatkan rancangan *Sequence diagram* pengembalian buku yang ditunjukkan pada Gambar 3.9.



**Gambar 3.9 *Sequence* Diagram Pengembalian Buku**

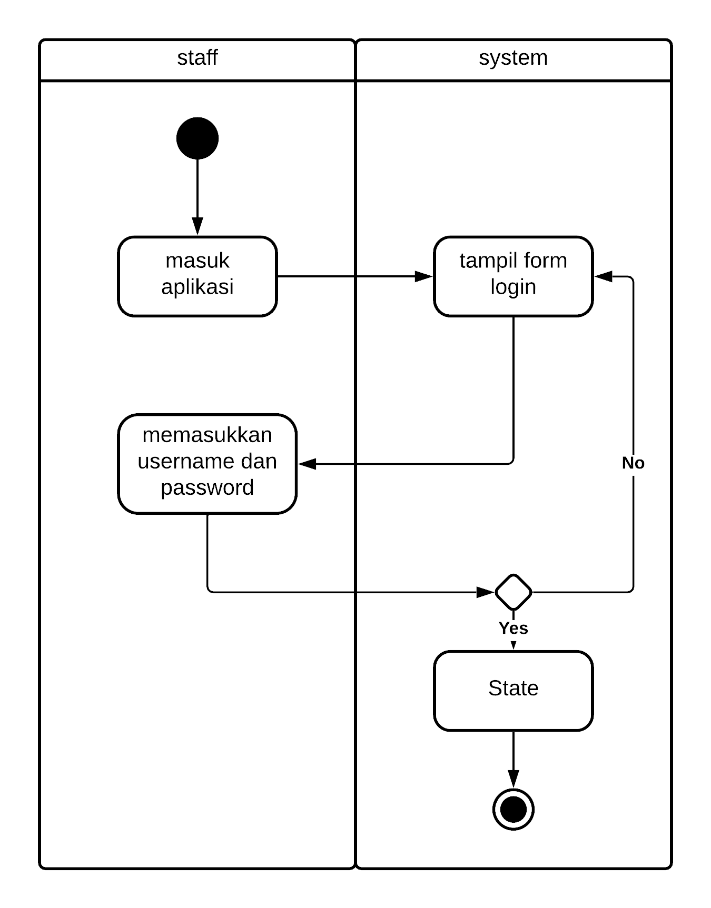
Adapun alur kerja dari *sequence* diagram pengembalian buku adalah sebagai berikut:

1. Penyewa mengembalikan buku, kemudian *user* mengecek kondisi dan tanggal pengembalian buku. Jika ada kerusakan dan keterlambatan pengembalian, *system* akan mencatat kerusakan dan keterlambatan tersebut ke *menu* arsip pinjam*.*
2. *User* membuat kwitansi denda untuk penyewa, dan menyimpan arsip kwitansi denda di *menu* arsip kwitansi.
   * 1. **Activity Diagram**

*Activity Diagram* merupakan rancangan aliran aktivitas atau aliran kerja dalam sebuah sistem yang akan dijalankan. *Activity Diagram* digunakan untuk mendefinisikan atau mengelompokan aliran tampilan dari sistem tersebut.

1. *Activity Diagram Login*

Setelah melakukan analisa sistem, maka didapatkan rancangan *activity diagram login* yang ditunjukkan pada Gambar 3.10.

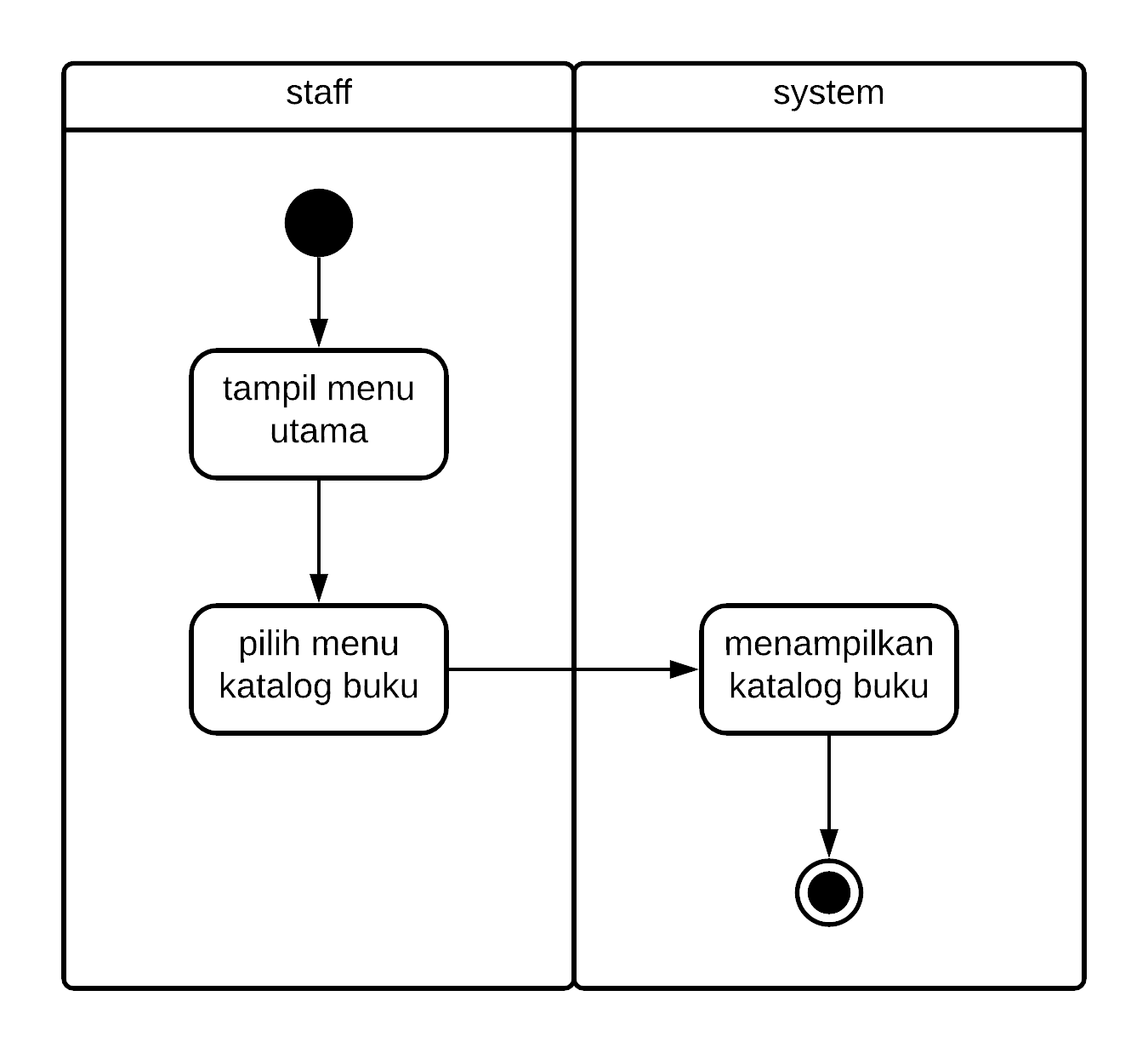


**Gambar 3.10 *Activity Diagram Login***

Adapun alur kerja dari *Activity diagram login* adalah sebagai berikut :

1. *User* masuk ke *website*/aplikasi.
2. Selanjutnya, sistem akan menampilkan *form login*.
3. *User* memasukkan *username* dan *password*. Jika *username* dan *password* yang dimasukkan tidak sesuai dengan *database* maka akan kembali ke *user* untuk memasukkan kembali *username* dan *password*. Jika *username* dan *password* yang dimasukkan sesuai maka *login* berhasil dan *user* dapat mengakses sistem.
4. *Activity Diagram* Melihat Katalog

Setelah melakukan analisa sistem, maka didapatkan rancangan *activity diagram* melihat katalog yang ditunjukkan pada Gambar 3.11.

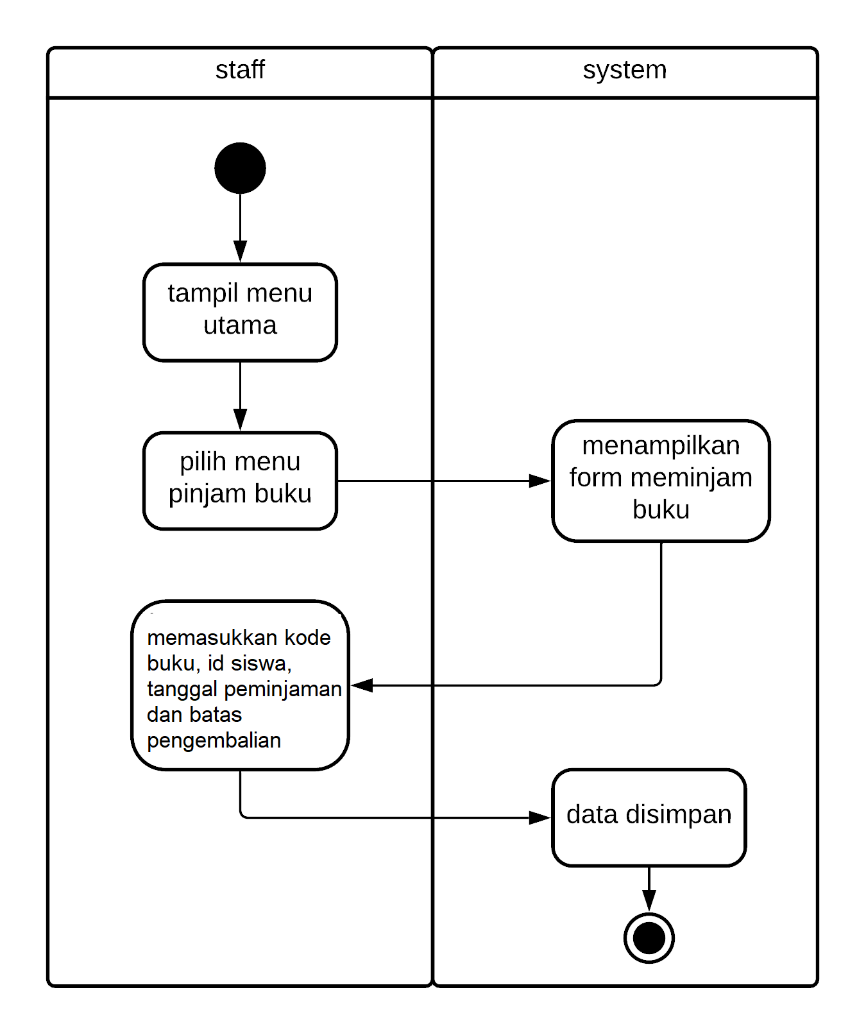


**Gambar 3.11 *Activity Diagram* Melihat Katalog**

Adapun alur kerja dari *Activity diagram* melihat katalog adalah sebagai berikut :

1. *User* memilih *menu* katalog buku.
2. *System* akan menampilkan tabel katalog buku
3. *Activity Diagram* Pinjam Buku

Setelah melakukan analisa sistem, maka didapatkan rancangan *activity diagram* pinjam buku yang ditunjukkan pada Gambar 3.12.

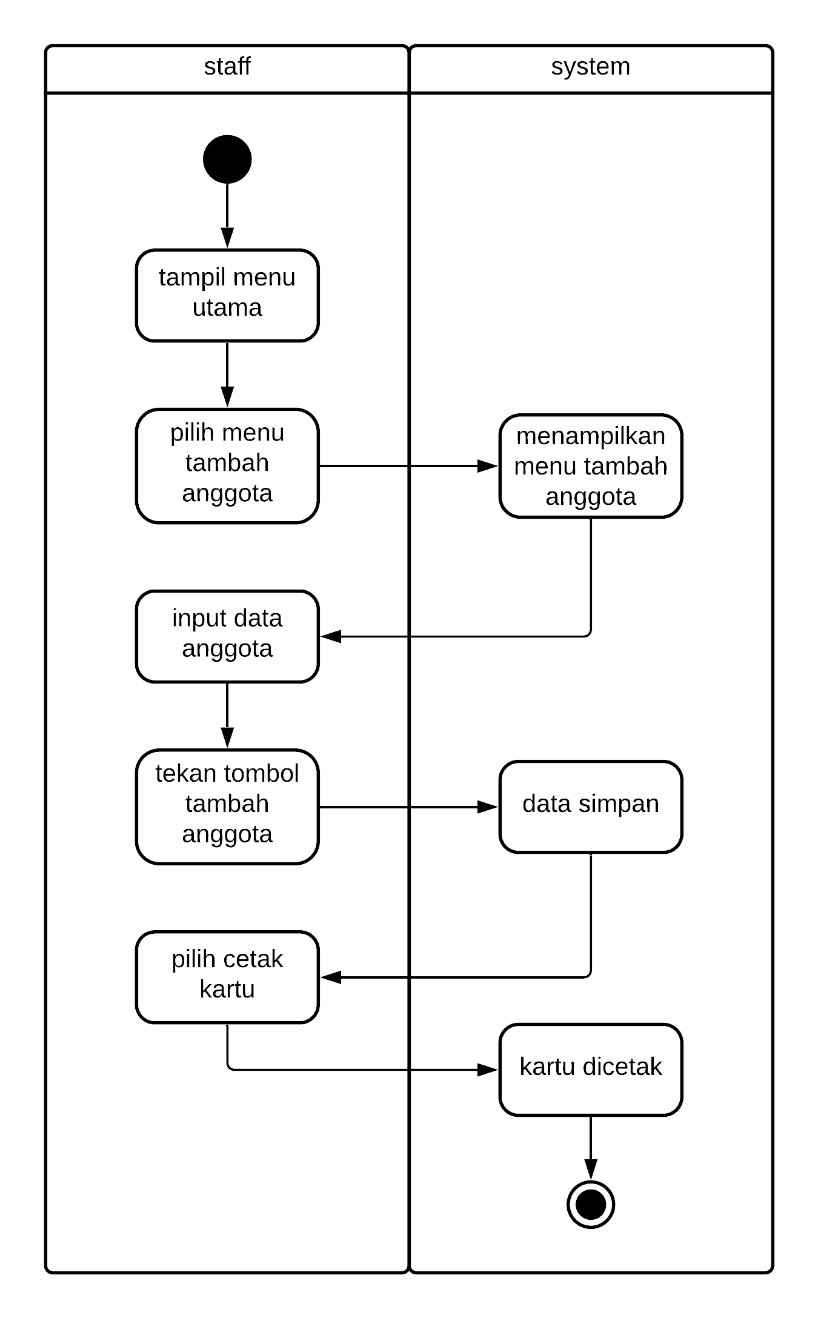


**Gambar 3.12 *Activity Diagram* Pinjam Buku**

Adapun alur kerja dari *Activity diagram* pinjam buku adalah sebagai berikut :

1. *User* memilih *menu* pinjam buku.
2. *System* akan menampilkan form peminjaman buku.
3. *User* menginput data buku dipinjam dan anggota yang memimjam buku.
4. Selanjutnya, *system* akan menyimpan data.
5. *Activity Diagram* Tambah Anggota

Setelah melakukan analisa sistem, maka didapatkan rancangan *activity diagram* tambah anggota yang ditunjukkan pada Gambar 3.13.



**Gambar 3.13 *Activity Diagram* Tambah Anggota**

Adapun alur kerja dari *Activity diagram* tambah anggota adalah sebagai berikut :

1. *User* memilih *menu* tambah anggota.
2. *System* menampilkan *menu* tambah anggota. Kemudian *user* menginput data anggota dan menekan tombol tambah anggota.
3. *System* akan menyimpan data.
4. Selanjutnya,  *user* memilih cetak kartu dan kartu dicetak.
5. *Activity Diagram* Tambah Buku

Setelah melakukan analisa sistem, maka didapatkan rancangan *activity diagram* tambah buku yang ditunjukkan pada Gambar 3.14.

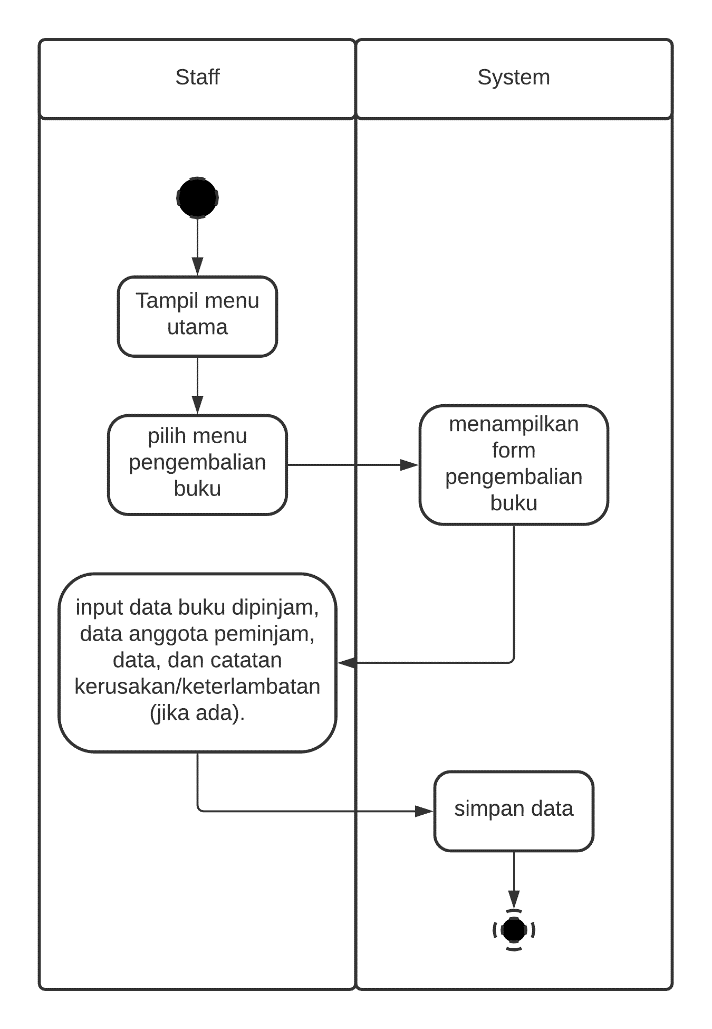


**Gambar 3.14 *Activity Diagram* Tambah Buku**

Adapun alur kerja dari *Activity diagram* tambah buku adalah sebagai berikut :

1. *User* memilih *menu* tambah buku.
2. *System* menampilkan *menu* tambah buku. Kemudian *user* menginput data buku.
3. *System* akan menyimpan data.
4. Selanjutnya, *user* memilih cetak stiker buku dan stiker dicetak.
5. *Activity Diagram* Pengembalian Buku

Setelah melakukan analisa sistem, maka didapatkan rancangan *activity diagram* pengembalian buku yang ditunjukkan pada Gambar 3.16.



**Gambar 3.15 Activity Diagram Pengembalian Buku**

Adapun alur kerja dari *activity diagram* pengembalian buku adalah sebagai berikut :

1. *User* memilih *menu* pengembalian buku.
2. *System* menampilkan *form* pengembalian buku
3. *User* menginput data buku yang dipinjam, data anggota peminjam dan catatan kerusakan/keterlambatan pengembalian buku jika ada.
4. *System* lalu menyimpan data.
   * 1. **Perancangan *User Interface***
5. Tampilan Halaman *Form Login*

Pada tampilan halaman *form login* ini terdapat *username* dan *password* yang harus diisi admin terlebih dahulu sebelum dapat masuk ke sistem dan mengakses sistem. Tampilan halaman *form login* dapat ditunjukkan pada Gambar 3.16.



**Gambar 3.16 Tampilan Halaman *Form Login***

1. Tampilan Halaman *Menu Home*

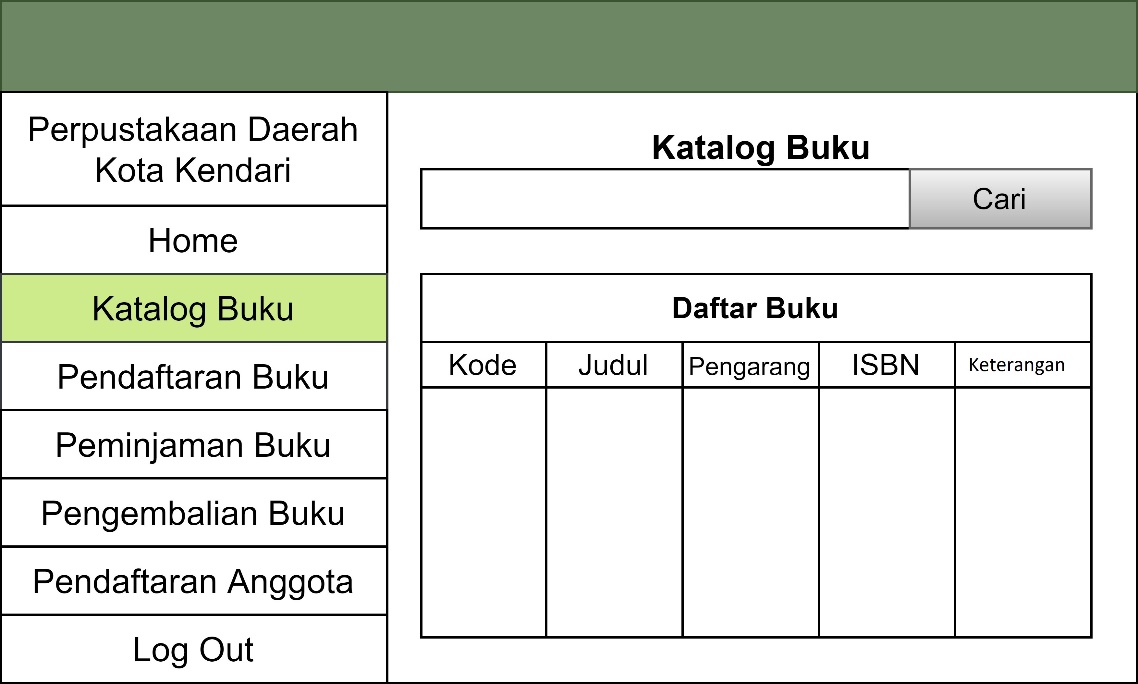
Tampilan halaman menu *home* beranda ditunjukkan pada Gambar 3.17.



**Gambar 3.17 Tampilan Halaman *Menu Home***

1. Tampilan Halaman *Menu* Katalog Buku

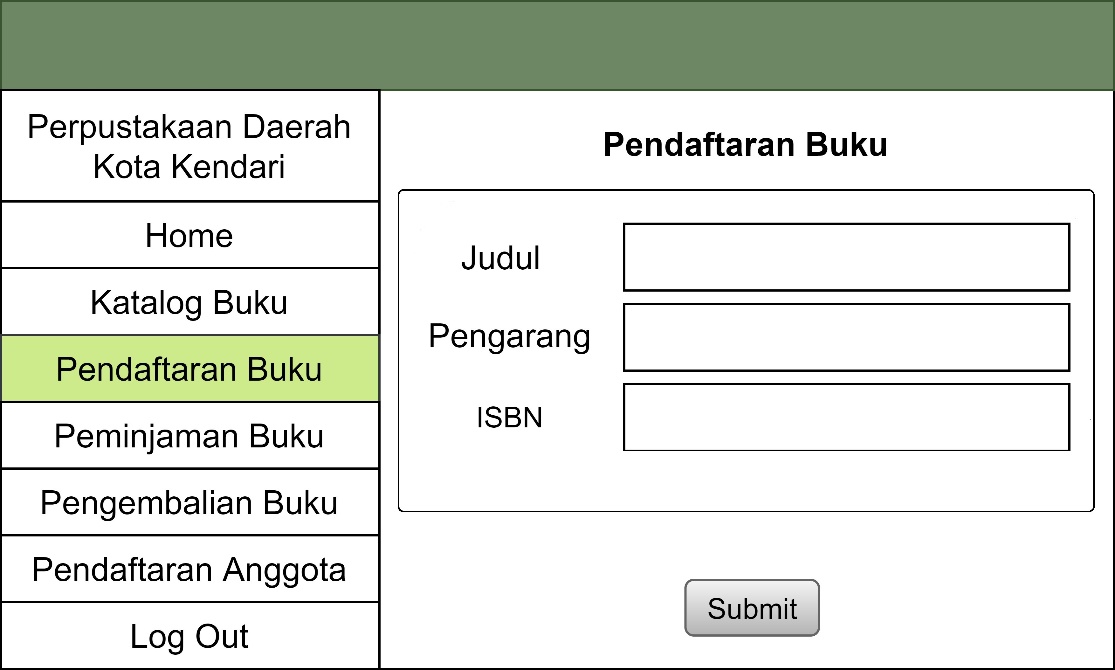
Pada tampilan halaman *menu* katalog buku terdapat kolom pencarian, daftar buku berupa kode, judul buku, pengarang, ISBN dan keterangan. Tampilan halaman menu katalog buku ditunjukkan pada gambar 3.18.



**Gambar 3.18 Tampilan *Menu* Katalog Buku**

1. Tampilan Halaman *Menu* Pendaftaran Buku

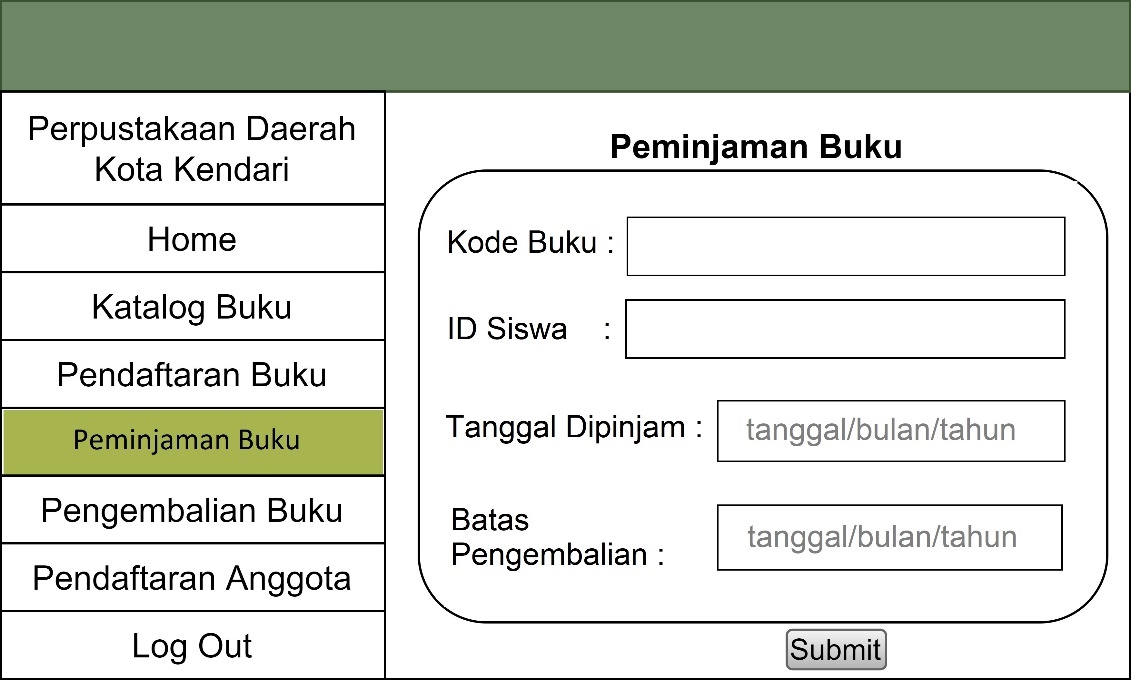
Pada tampilan halaman *menu* pendaftaran buku terdapat form pengisian judul, pengarang, dan ISBN. Tampilan halaman *menu* pendaftaran buku ditunjukkan pada gambar 3.19.



**Gambar 3.19 Tampilan Halaman *Menu* Pendaftaran Buku**

1. Tampilan *Menu* Peminjaman Buku

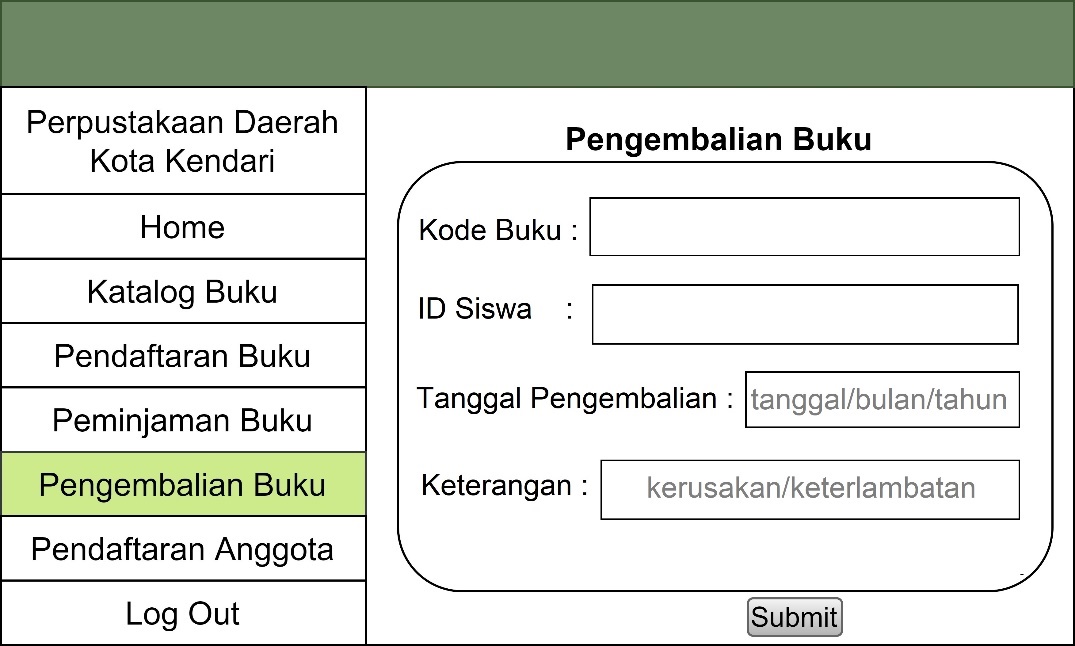
Pada Tampilan *menu* peminjaman buku terdapat *form* pengisian kode buku, id siswa, tanggal dipinjam, dan batas pengembalian. Tampilan *menu* peminjaman buku ditunjukkan pada gambar 3.20.



**Gambar 3.20 Tampilan *Menu* Peminjaman Buku**

1. Tampilan *Menu* Pengembalian Buku

Pada tampilan *menu* pengembalian buku terdapat *form* pengisian kode buku, id siswa, tanggal pengembalian, dan keterangan kerusakan/keterlambatan. Tampilan *menu* pengembalian buku ditunjukkan pada gambar 3.21.



**Gambar 3.21 Tampilan *Menu* Pengembalian Buku**

1. Tampilan *Menu* Pendaftaran Anggota

Pada tampilan *menu* pendaftar an anggota terdapat *form* pengisian nama dan alamat. Tampilkan *menu* pendaftaran anggota ditunjukkan pada gambar 3.22.



**Gambar 3.22 Tampilan *Menu* Pendaftaran Anggota**

**3.6. Pengujian**

* + 1. **Pengujian *Black Box***

Pengertian dari *black box* testing merupakan teknik pengujian yang berfokus pada spesifikasi fungsional dari perangkat lunak. Pengujian bekerja dengan mengabaikan struktur pada control sehingga berfokus pada informasi domain. Pengujian menggunakan *black box* testing memungkinkan pengembang sistem untuk membuat himpunan kondisi input yang akan melatih seluruh batasanbatasan fungsional pada suatu system (Jaya, 2018).

Keuntungan dalam menggunakan metode *black box* testing adalah pengujian tidak perlu memiliki pengetahuan tentang bahasa pemrograman tertentu. Pengujian dilakukan dari sudut pandang pengguna sehingga programmer dan tester keduanya saling bergantung satu sama lain (Hidayat dan Putri, 2019).

* + 1. **Pengujian Efisiensi Waktu**

Salah satu pengujian yang akan dilakukan yaitu pengujian efisiensi waktu. Efisien dapat diartikan sebagai cara untuk mencapai suatu tujuan yang optimal (cepat dan tepat) serta sesuai keinginan, dengan meminimalkan sumber daya yang dikeluarkan. Sumber daya yang dimaksud adalah tenaga, uang, dan waktu (Dicoding Intern, 2021). Pencarian dikatakan valid jika waktu yang dibutuhkan dalam mencari sebuah kode buku sangat singkat.

**DAFTAR PUSTAKA**

Abdurrahman, J. dan No, S. (2016) “APLIKASI KAMUS BAHASA INDONESIA – BUGIS BERBASIS WEB DENGAN METODE SEQUENTIAL SEARCH,” 3(September), hal. 246–258.

Dicoding Intern (2021) *Apa Itu Efisien? Pengertian dan Contohnya*, *28 April*. Tersedia pada: https://www.dicoding.com/blog/apa-itu-efisien-pengertian-dan-contohnya/.

dosenpendidikan (2020) *Pengertian Buku*, *28/12/2020*. Tersedia pada: https://www.dosenpendidikan.co.id/pengertian-buku/.

Ferdianto, Y. (2013) “Pengertian PHP.”

H. KAZWINI (2010) *SEARCHING : INTERPOLATION SEARCH*, *Desember 22, 2010*. Tersedia pada: https://kazwini13.wordpress.com/2010/12/22/searching-interpolation-search/#:~:text=Interpolation Search adalah sebuah algoritma,kunci yang terdapat pada buku.

Heri (2017) *SIMBOL FLOWCHART : Pengertian, Jenis, Fungsi dan Contohnya*.

Hidayat, T. dan Putri, H. D. (2019) “Pengujian Portal Mahasiswa pada Sistem Informasi Akademik ( SINA ) menggunakan Black Box Testing dengan Metode Equivalence Partitioning dan Boundary Value Analysis,” *Jurnal Teknik Informatika UNIS (JUTIS)*, 7(1), hal. 83–92.

Ii, B. A. B., Tentang, A. H. dan Teks, B. (2012) “BAB II\_KHAFIF ALI AKBAR\_PPKn’16,” hal. 12–34.

Kiswanto, C. F. (2016) “‘Sistem Informasi Akademik Sub-Sistem : Utility Dan Epsbed.’”

Lestari, I. D. (2016) “Klasifikasi online dan google, Jurnal Iqra’ Volume 10 No.02,” hal. 83–94.

M. Roziq Zanuddin, H. A. R. S. (2016) “Sistem Informasi Perpustakaan Umum Grati Kabupaten Pasuruan Berbasis Web Menggunakan Program Php Dan Database *MySQL*,” *J I M P - Jurnal Informatika Merdeka Pasuruan*, 1(2), hal. 1–14. doi: 10.37438/jimp.v1i2.13.

Paira, S. dan Chandra, S. (2014) “*Bi Linear* Search a New Session of Searching International Journal of Advanced Research in *Bi Linear* Search a New Session of Searching,” (May 2015).

Saputro, B. I. (2017) “Penerapan Sistem Klasifikasi Perpustakaan Arkeologi di Perpustakaan Balai Arkeologi Daerah Istimewa Yogyakarta,” *Berkala Ilmu Perpustakaan dan Informasi*, 13(2), hal. 107. doi: 10.22146/bip.23453.

Sataloff, R. T., Johns, M. M. dan Kost, K. M. (2017) “OPTIMALISASI SISTEM PENCARIAN DATA BUKU UNTUK PENGAMBILAN KEPUTUSAN DI PERPUSTAKAAN.”

Seidl, T. dan Enderle, J. (2011) “Binary search,” *Algorithms unplugged*, hal. 5–11. doi: 10.1007/978-3-642-15328-0\_1.

Sembiring, J. P. (2013) “PERANCANGAN APLIKASI KAMUS BAHASA INDONESIA - KARO ONLINE BERBASIS WEB DENGAN METODE SEQUENTIAL SEARCH,” hal. 28–33.

Setiawan Dimas (2019) *Pengertian CodeIgniter dan Konsep MVC (Model View Controller)*, *Kelas Programmer*.

Sultana, N. *et al.* (2017) “A brief study and analysis of different searching algorithms,” *Proceedings of the 2017 2nd IEEE International Conference on Electrical, Computer and Communication Technologies, ICECCT 2017*. doi: 10.1109/ICECCT.2017.8117821.

Watkins, J. (2001) “Testing IT An Off The Shelf Software Testing Process,” *Jurnal Informatika Pengembangan IT (JPIT)*, 3(2), hal. 45–46.

Wulan Nafesa Septine, S.T., M. (2019) *DATABASE MANAGEMENT SYSTEM (DBMS) MARIA DB*, *06 Nov 2019*.